Mitral Regurgitation Assessment System for Contrast Two-Dimensional Echocardiograms

Zhang Shuqing (张蜀青)

July - 1991
Mitral Regurgitation Assessment System for Contrast Two-Dimensional Echocardiograms

Ph.D. Dissertation in Biomedical Engineering
submitted to the Faculty of Engineering, University of Oporto

Zhang Shuqing (张蜀青)

B. Sc. in Radio Electronics
Sichuan University
Chengdu, the P.R. China, 1983

M. Eng. in Biomedical Engineering
Graduate School of University of Science and Technology of China
Beijing, the P. R. China, 1986

July - 1991
Mitral Regurgitation Assessment System for Contrast Two-Dimensional Echocardiograms

P.D. Dissertation in Biomedical Engineering

Submitted to the Faculty of Engineering, University of Porto

Xiang Shuiping

UNIVERSIDADE DO PORTO
Faculdade de Engenharia
BIBLIOTECA

N.° 34291
CDU 681.3(043)
Data 13-2-2001
Dedicated to

my beloved parents

whom I owe so much
Acknowledgements

Support for this work was provided by the JNICT (Junta Nacional de Investigação Científica e Tecnológica) - under research grant No. 899.86.40/IMU, and by the Fundação Oriente. The preparation of this Ph.D. dissertation and the trip to Chicago for presentation of a paper in the International Conference of Computers in Cardiology'90 were also funded by the Fundação Oriente.

Prof. Joaquim Pontes Marques de Sá, from the Faculty of Engineering, University of Oporto, acted as principal supervisor. He gave me three invaluable years of collaboration and friendship. His warm personality, keen insight, and eye for detail have shaped me as a scientist and as a person.

Prof. Marques de Sá has also helped me to clarify many of the ideas in this dissertation as well as in my publications, and worked with me to make the presentation more focused and more understandable.

Prof. Cassiano Abreu-Lima, from the Faculty of Medicine, University of Oporto, acted as medical supervisor. He helped me to understand the medical problems in this work and guided me to new frontiers of cardiology. Besides, I would also be grateful to him for his helpful comments on medical aspects in this dissertation.

Dr. Manuel Guerreiro, from the "Hospital de São João", collaborated with me in this work. I am grateful to him for his preparation of the contrast echocardiograms on video cassettes and for some references I requested. In addition, I also appreciate his patience to discuss and solve problems I had with video images.

I would like to thank Prof. A. Campilho and Eng. A. M. P. Monteiro, both from the Faculty of Engineering, University of Oporto, for their help and constructive suggestions during the development of this work.

I also thank all the colleagues, friends, and staffs who offered me their help any time I needed.
Contents

Notations

1. Introduction
   1.1. Conventional Methods in MR Assessment
      1.1.1. Cineangiography
      1.1.2. Doppler Echocardiography
      1.1.3. Contrast Echocardiography
   1.2. MR Assessment by Contrast 2-D Echocardiography
      1.2.1. Preparation of Contrast 2-D Echocardiograms
      1.2.2. Evaluation Criteria
      1.2.3. Clinical Validation of MR Assessment
   1.3. Contributions of the Present Work

2. System Overview
   2.1. Hardware System Configuration
      2.1.1. Functional Structure and Main Characteristics
      2.1.2. Image Acquisition and Processing Boards
   2.2. Software System Configuration
      2.2.1. Image Acquisition
      2.2.2. Regions of Interest Definition
      2.2.3. Microbubble Detection
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.4. Microbubble Motion Parameter Extraction  23</td>
</tr>
<tr>
<td>2.2.5. MR Feature Selection  23</td>
</tr>
<tr>
<td>2.2.6. Linear Classification  24</td>
</tr>
<tr>
<td>3. Contrast Echocardiogram Processing  25</td>
</tr>
<tr>
<td>3.1. Sequential Image Acquisition and Alignment  25</td>
</tr>
<tr>
<td>3.1.1. Image Acquisition  26</td>
</tr>
<tr>
<td>3.1.1.1. Number of Image Frames  26</td>
</tr>
<tr>
<td>3.1.1.2. Image Files  29</td>
</tr>
<tr>
<td>3.1.2. Temporal Alignment of Image Frames  30</td>
</tr>
<tr>
<td>3.1.2.1. Manual Control Alignment  30</td>
</tr>
<tr>
<td>3.1.2.2. Automatic Control Alignment  35</td>
</tr>
<tr>
<td>3.1.3. Spatial Alignment of Cardiac Chambers  36</td>
</tr>
<tr>
<td>3.1.3.1. Image Matching Algorithm  38</td>
</tr>
<tr>
<td>3.1.3.2. Automatic Spatial Alignment  39</td>
</tr>
<tr>
<td>3.1.3.3. Image Matching Realization  41</td>
</tr>
<tr>
<td>3.1.3.3.1. Matching Region Definition  41</td>
</tr>
<tr>
<td>3.1.3.3.2. Binary Images Operation  43</td>
</tr>
<tr>
<td>3.2. Regions of Interest Definition  45</td>
</tr>
<tr>
<td>3.2.1. ROIs Definition in Contrast-Free Images  46</td>
</tr>
<tr>
<td>3.2.2. ROIs Adjustment in Contrast Images  55</td>
</tr>
<tr>
<td>3.3. Detection of Microbubbles  59</td>
</tr>
<tr>
<td>3.3.1. Microbubbles Detection Based on their Characteristics  59</td>
</tr>
<tr>
<td>3.3.2. Linear Relationship Assumptions  61</td>
</tr>
<tr>
<td>3.3.3. Linear Regressive Relationship  64</td>
</tr>
<tr>
<td>3.3.4. Adaptive Detection of Microbubbles  70</td>
</tr>
<tr>
<td>Contents</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>4. MR Pattern Recognition</td>
</tr>
<tr>
<td>4.1. General Theory of Pattern Recognition</td>
</tr>
<tr>
<td>4.4.1. Pattern Recognition Process</td>
</tr>
<tr>
<td>4.4.2. Basic Methodologies of Pattern Recognition</td>
</tr>
<tr>
<td>4.4.3. Implementation of Pattern Recognition</td>
</tr>
<tr>
<td>4.4.4. MR as a Pattern Recognition Problem</td>
</tr>
<tr>
<td>4.2. Microbubble Motion Patterns Analysis</td>
</tr>
<tr>
<td>4.2.1. MR Criteria Interpretation</td>
</tr>
<tr>
<td>4.2.2. Basic Microbubbles Measures</td>
</tr>
<tr>
<td>4.2.3. Microbubble Motion Measurements</td>
</tr>
<tr>
<td>4.3. Theory of Classification Rule</td>
</tr>
<tr>
<td>4.3.1. Bayes Linear Classification Rule</td>
</tr>
<tr>
<td>4.3.2. Hierarchical Linear Classification Rule</td>
</tr>
<tr>
<td>4.4. MR Feature Selection</td>
</tr>
<tr>
<td>4.4.1. Previous Studies on Feature Selection</td>
</tr>
<tr>
<td>4.4.2. Sample Size and Feature Size</td>
</tr>
<tr>
<td>4.4.3. Feature Selection by $F$ Statistic</td>
</tr>
<tr>
<td>4.4.4. Classification Tree Selection</td>
</tr>
<tr>
<td>4.4.5. Features for the Classifier</td>
</tr>
<tr>
<td>4.4.5.1. Interpretation of Features at Node 1</td>
</tr>
<tr>
<td>4.4.5.2. Interpretation of Features at Node 2</td>
</tr>
<tr>
<td>4.4.5.3. Interpretation of Features at Node 3</td>
</tr>
<tr>
<td>4.5. Classification Results and Validation</td>
</tr>
<tr>
<td>4.5.1. Classification Results in Training Set</td>
</tr>
<tr>
<td>4.5.2. Estimation Methods of Classifier Performance</td>
</tr>
<tr>
<td>4.5.3. Rotation Method</td>
</tr>
<tr>
<td>4.5.4. Estimation Results in Test Sets</td>
</tr>
</tbody>
</table>
### MIRAS for Contrast 2-D Echocardiograms

<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5.5. Examples of Classification</td>
<td>122</td>
</tr>
<tr>
<td>4.5.5.1. Example 1</td>
<td>122</td>
</tr>
<tr>
<td>4.5.5.2. Example 2</td>
<td>123</td>
</tr>
<tr>
<td>4.5.5.3. Example 3</td>
<td>123</td>
</tr>
</tbody>
</table>

### 5. Conclusion

125

### References

129

### Appendix I. Program Lists

- i. Sequential Image Acquisition and Alignment Programs
- ii. ROI Definition Programs
- iii. Features Calculation and Case Classification Programs
- iv. Training Database Management Programs

### Appendix II. MIRAS Operation Phases

- i. File Names and Structures
- ii. Operation Phases
- iii. Main Screens of Performance

### Appendix III. DT2851, DT2858 and DT-IRIS

- i. DT2851 Technical Data
- ii. DT2858 Technical Data
- iii. DT-IRIS Summary
Notations

The following abbreviations are used in this dissertation.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-D</td>
<td>2-Dimensional</td>
</tr>
<tr>
<td>AML</td>
<td>Anterior Mitral Leaflet</td>
</tr>
<tr>
<td>DT</td>
<td>Diastole</td>
</tr>
<tr>
<td>ED</td>
<td>End Diastole</td>
</tr>
<tr>
<td>ES</td>
<td>End Systole</td>
</tr>
<tr>
<td>LA</td>
<td>Left Atrium</td>
</tr>
<tr>
<td>LSB</td>
<td>Least Significant Bit</td>
</tr>
<tr>
<td>LV</td>
<td>Left Ventricle</td>
</tr>
<tr>
<td>MIRAS</td>
<td>Mitral Regurgitation Assessment System</td>
</tr>
<tr>
<td>MR</td>
<td>Mitral Regurgitation</td>
</tr>
<tr>
<td>PML</td>
<td>Posterior Mitral Leaflet</td>
</tr>
<tr>
<td>ROI</td>
<td>Region of Interest</td>
</tr>
<tr>
<td>ROIs</td>
<td>Regions of Interest</td>
</tr>
<tr>
<td>SEN</td>
<td>Sensitivity</td>
</tr>
<tr>
<td>SPE</td>
<td>Specificity</td>
</tr>
<tr>
<td>ST</td>
<td>Systole</td>
</tr>
</tbody>
</table>
1. Introduction

Since 1968, when Gramiak and Shah [1] first reported on the enhancement of ultrasound image by injecting idocyanine green contrast microbubble agents in the heart during echocardiography, the contrast echo techniques have generally been used to identify valve regurgitation, intracardiac shunts and cardiac chambers themselves. They have become an alternative choice to X-ray angiography [2].

Valvular regurgitation has been extensively studied in some laboratories by using hand-agitated microbubbles. Although valvular regurgitation can be evaluated with some existing methods, such as cineangiography, Doppler echocardiography, etc., previous studies have shown that contrast two-dimensional (2-D) echocardiography for the assessment of valvular regurgitation has some advantages. In this thesis, we intend to discuss the evaluation of mitral regurgitation (MR) in the operating room by contrast 2-D echocardiography and its automated quantitative analysis and interpretation. The system we developed for this particular purpose is, to the best of our knowledge, the first that has been reported.

1.1. Conventional Methods in MR Assessment

1.1.1. Doppler Echocardiography

MR can be detected intraoperatively by left atrial hemodynamic monitoring, direct visual inspection, digital palpation or dye injections, but these methods may be inaccurate and underestimate the presence and severity of the regurgitation [3]. There are also some conventional efficient techniques widely used in operating rooms, such as cineangiography, Doppler echocardiography, etc., but they have been found to exhibit some limitations when applied in intraoperative MR assessment.
1.1.1. Cineangiography

Cineangiography is used widely for the assessment of the presence and severity of valvular regurgitation \cite{4} - \cite{7}. There are, however, some disadvantages in cineangiography.

Contrast media used in angiography may cause peripheral vasodilatation, increased intravascular volume and increased cardiac output which can result in a significant clinical problem to the patients when their left ventricular function has been impaired because of ischemic or valvular heart disease \cite{8}. In patients whose left ventricular function is already compromised because of ischemic or valvular heart disease, these physiologic alterations may pose a significant clinical problem \cite{8}. Besides, contrast-induced renal failure may be precipitated, particularly in patients with preexisting renal disease \cite{9} and severe allergic reactions may occur \cite{10}. Thus, the use of contrast dye may not be desirable in some selected patients.

In addition, cineangiography in the situation of multiple views is limited by the dosage of contrast medium that may be used in an individual patient and by the overlapping of contrast-filled structures. The radiation exposure of the cineangiography can occasionally become a significant limitation to its use, especially in pregnant women, when adverse effects of the radiation to the fetus must be considered \cite{11} \cite{12}. Moreover, the semiquantitative assessment by cineangiography has some technical problems such as premature ventricular contractions, particularly in left ventriculography, which may prevent accurate evaluation of left ventricular function and the detection and quantification of MR \cite{5}.

1.1.2. Doppler Echocardiography

Conventional pulsed Doppler echocardiography has been used intraoperatively to evaluate the presence and severity of MR \cite{13} by detecting the turbulent flow in the left atrium (LA). However, this technique has been known to both underestimate and overestimate the severity of MR \cite{14}, and it requires great skill and time to make the Doppler examination. Besides, the limited view of spatial orientation can also prevent its application in the operating room.
Color Doppler flow imaging is an important technical innovation that combines color-coded Doppler-derived flow data with either M-mode or 2-D echocardiograms. Previous studies have shown that this technique is highly sensitive and specific, compared with angiography, in the detection of MR presence \cite{15} - \cite{17} and useful in the accurate estimation of its severity \cite{17}. In addition, color Doppler echocardiography facilitates the visualization of mitral regurgitant flow, takes much less time in the examination than the conventional pulsed Doppler technique and demonstrates well the spatial orientation and the extent of the mitral regurgitant jet. However, its relatively expensive equipment largely restrains it from being utilized in wide range.

\subsection{1.1.3. Contrast Echocardiography}

Intraoperative contrast 2-D echocardiography using hand-agitated microbubbles generated from low cost contrast materials (saline solutions) has been recognized to be an innocuous method which permits multiple injections in different echocardiographic views to assess complex intracardiac flow patterns \cite{5}. Intraoperative contrast 2-D echocardiography has been found to be highly sensitive and specific in the detection of mitral valve regurgitation compared with cineangiography. Its assessment based on a visual grading scale has an excellent correlation with that obtained by cineangiography \cite{5} \cite{7}. Therefore, it has been used to evaluate intraoperatively mitral valve operations \cite{18} \cite{19}.

Although contrast M-mode echocardiography has been used during cardiac catheterization to detect MR and aortic regurgitation (AR) \cite{20}, there exist some disadvantages of evaluating the severity of the regurgitant lesions by M-mode echocardiography because of the limited view of the intracardiac chambers. Contrast 2-D echocardiography, on the contrary, is particularly useful because of the added spatial orientation it provides, and therefore has been widely used.
1.2. MR Assessment by Contrast 2-D Echocardiography

1.2.1. Evaluation Criteria

The previous reports mentioned above have demonstrated that intraoperative contrast 2-D echocardiography for the evaluation of the presence of MR is excellently sensitive and specific when compared with semiquantitative cineangiography and that it could accurately quantify the severity of left-sided valvular regurgitation [5] - [7]. Meanwhile, Goldman ME et al. [7] also compared the results of tricuspid regurgitation assessment by intraoperative echocardiography and preoperative pulsed Doppler echocardiography and showed its high sensitivity and specificity.

1.2.1. Preparation of Contrast 2-D Echocardiograms

Intraoperative contrast 2-D echocardiography was performed on 157 patients submitted to open heart surgery at "Hospital de São João", Oporto, Portugal, between April 1986 and 1990, by medical doctors using an echocardiograph ALOKA, model SSD-720, with sectorial mechanical 5MHz transducer, model ASU-28S-5. Pulsed Doppler echocardiography was performed before operation as a standard method on those patients for comparison studies.

During the thoracic operation, the prepared hand-agitated microbubbles of a sodium chloride solution mixed with the patient's blood was rapidly injected through intracardiac catheter, which was connected to a three-way stopcock, into the left ventricle (LV) before and after mitral valve surgery, in mitral regurgitant patients, for the assessment of MR and of the mitral valve operation results. In some patients, aortic regurgitation and tricuspid regurgitation were also studied with the injections in the aortic root and the right ventricle, respectively.

Basal images obtained before contrast injection and images with contrast microbubbles of both long- and short-axis views were recorded on VHS video cassettes for later analysis. The injections and recordings were repeated in most of the patients for obtaining satisfactory images because of the existence of some technical problems. These problems might happen when lower quantity contrast agent was injected, the brightness or contrast of the echocardiograph was not well adjusted, the cardiac imaging plane did not catch the microbubble region, or cardiac chamber image is too small, etc.
1.2.2. Evaluation Criteria

As it is from common knowledge, the blood in a healthy heart flows to LV from LA during systole (ST), and then flows to aorta from LV during diastole (DT) without coming back to LA. When mitral regurgitation exists, the blood in LV not only flows to aorta, but also back to LA during ST.

The patient images were evaluated visually by a physician. The evaluation was performed on the sequences of echocardiograms, starting with the basal frame and continuing with those obtained after the injection of contrast agent in LV, analyzing the evolution of the contrast material in the LV and comparing its concentration with that of the contrast material eventually present in LA, which is then indicative of MR [7][21][13][19].

The presence and severity of MR were then categorized in the following classes:

Class 0 (no regurgitation): No microbubbles in LA.

Class 1 (mild): Only a few microbubbles appear in the LA and a jet effect can be observed. The jet effect makes microbubbles concentrated in a certain region in LA which is caused by small orifices in the mitral valve.

Class 2 (moderate): The microbubbles completely fill the LA in the first two systoles, but less densely than in the LV; a jet effect can also be observed.

Class 3 (severe): The whole LA region is filled with microbubbles from the first to nearly the tenth systole and with a density progressively greater than or at least equal to the density in LV with each heart beat.

Typical examples of the different class images are shown in Figures 1-4.
Figure 1. An example of the negative class image (class 0); there are no or few microbubbles present in LA.

Figure 2. An example of mild MR (class 1); there are few microbubbles in LA which concentrate in a certain region (jet effect).
Figure 3. An example of moderate MR (class 2); there are some microbubbles present in the whole LA region but with some inhomogeneous distribution caused by the "jet effect".

Figure 4. An example of severe MR (class 3); a lot of microbubbles can be observed in the whole LA with an almost homogeneous distribution.
1.2.3. Clinical Validation of MR Assessment

In order to develop the automated analysis and interpretation methods with heuristic approaches for the MR assessment in contrast 2-D echocardiograms, let us first review and analyze the visual assessment results obtained by Dr. Manuel Guerreiro in the "Hospital de São João", Oporto, to whom this work is much in debt, in what concerns clinical data, classification criteria and medical knowledge.

The efficiency of the MR assessment by intraoperative contrast 2-D echocardiography was validated by Dr. Guerreiro [13]. He compared the visual assessment results obtained by the intraoperative contrast 2-D echocardiography and by pulsed Doppler echocardiography. Based on the above criteria and his clinical experience, the pre- and post-operative contrast 2-D echocardiographic images of 157 patients were visually classified in the four classes. The MR of the 157 patients was also visually evaluated in those four classes by pulsed Doppler echocardiography before mitral valve surgery. Both results are listed in the matrix of Table 1.

<table>
<thead>
<tr>
<th>Contrasted 2-D Echocardiography</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulsed Doppler Echocardiography</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>98</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>23</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 1. Results of semiquantitative MR assessment by contrast 2-D echocardiography using pulsed Doppler echocardiography as standard, the criteria and the physician's experience were both applied in the visual inspection. (Matrix based on clinical data kindly forwarded by Dr. Guerreiro)
For comparing the ability of a new method detecting negative and positive with a standard method, sensitivity and specificity of the new method are often used as important parameters. Sensitivity (SEN) and specificity (SPE) are estimated by using the following equations [22]:

\[
\text{SEN} \% = \frac{TP}{TP + FN} \\
\text{SPE} \% = \frac{TN}{TN + FP}
\]

(1.1) (1.2)

where TP is the number of true positives, FN the number of false negatives, TN the number of true negatives, and FP the number of false positives.

Compared with pulsed Doppler echocardiography, contrast 2-D echocardiography for detection of MR was estimated by the following parameters:

SEN: 95%
SPE: 99%

The correct classification by contrast 2-D echocardiography in the following classes were

<table>
<thead>
<tr>
<th>Class</th>
<th>Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0</td>
<td>99%</td>
</tr>
<tr>
<td>Class 1</td>
<td>77.8%</td>
</tr>
<tr>
<td>Class 2</td>
<td>92%</td>
</tr>
<tr>
<td>Class 3</td>
<td>100%</td>
</tr>
</tbody>
</table>

The average correct classification rate was 96.8%.

The clinical study results have therefore shown that contrast 2-D echocardiography is a reliable method for the detection of the presence of MR and for the assessment of its severity by an experienced physician with the existing criteria, and is easier to perform than pulsed Doppler echocardiography during operation.
However, actually the microbubbles presence in LA does not always represent MR, and vice versa. They may be caused by atrial fibrillation, holes in myocardial wall leaking microbubbles to LA, etc., but not by mitral valve defect. Besides of this, the jet effect which may appear in Class 1 and Class 2 may cause false-negative class appearance because the microbubbles are not always visible when they are jetted off the image plane.

Strictly speaking, the MR assessment results in Table 1 obtained from contrast 2-D echocardiography are not the exact interpretation of the established MR criteria. Additional information from the doctor’s experience of cardiac surgery was used. For example, he may judge a MR to be Class 0 by his previous surgical observation or by observing the mitral valve itself in the echocardiogram even when microbubbles appear in LA.

However, for the sake of an automated assessment tool design, we understood that the interpretation of contrast echocardiograms should only rely on its substantial criteria. In order to obtain an MR assessment database according only tightly to the mentioned existing criteria, Dr. Guerreiro and I reevaluated visually the contrast 2-D echocardiograms of the 157 patients, largely neglecting therefore the doctor’s experience.

<table>
<thead>
<tr>
<th>Pulsed Doppler Echocardiography</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>68</td>
<td>26</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 2. Results of semiquantitative MR assessment by contrast 2-D echocardiography using pulsed Doppler echocardiography as standard, when the criteria alone are applied in a straight way.
derived limits. The relationship between the results of assessment in 156 patients (the 157th patient image was not included because of too poor quality) by contrast 2-D echocardiography and pulsed Doppler echocardiography was obtained as in the following matrix (Table 2).

Compared with pulsed Doppler echocardiography, sensitivity and specificity, as well as correct classification rates were calculated again as follows:

- **SEN:** 97%
- **SPE:** 69%

The average correct classification rate was 76.3%, which is the same as that from Table 2. The correctly accepted visual assessment results given in Table 2, which were proven to be reliable by the biopsy, were shown to have high specificity and high classification rates. It can then be considered as a prototype choice for quantitative MR assessment in the contrast 2-D echocardiograms when no clinical criteria is used alone. In addition, the contrast echocardiography results in Table 2 give us a more satisfactory number of cases in each class.

Class 0: 69.4%
Class 1: 77.8%
Class 2: 80%
Class 3: 100%

The average correct classification rate was 76.3%.

Thus, the sensitivity of contrast 2-D echocardiography for detection of MR was still high (97%), but the specificity decreased now to 69%. The average correct classification rate was decreased to 76.3%.

As a matter of fact, there exists an ambiguity between adjacent classes in the visual assessment of MR by contrast 2-D echocardiography. The same mask covered cells in Table 2 can actually be considered as belonging to the class identical to that of pulsed Doppler echocardiography since there is no clinical loss of misclassification in the adjacent classes. Therefore, we obtain an partially correct assessment that is still found to be acceptable by the doctor in clinical applications.

In this way, contrast 2-D echocardiography for detection of MR, compared with pulsed Doppler echocardiography, still retains clinically acceptable high sensitivity and specificity, and also high correct classification rates as follows:
Introduction

SEN: 98%
SPE: 96%

Correct classification rates were

Class 0: 96%
Class 1: 100%
Class 2: 96%
Class 3: 100%

The average correct classification rate was 96.8% which is the same as that from Table 1.

The clinically acceptable visual assessment results given in Table 2, which were proved to be reliable by the high sensitivity, high specificity, and high classification rates, can thus be considered as a good database for quantitative MR assessment in the contrast 2-D echocardiograms when the MR criteria is used alone. In addition, the contrast echocardiography results in Table 2 give us a more satisfactory number of cases in each class than in Table 1 for an automated classifier design.

Finally, we can, therefore, conclude that this safe and reproducible technique, which is easy to learn and teach, is rapid and highly reliable in the intraoperative detection and evaluation of MR, and is an attractive low cost alternative to color Doppler echocardiography.

1.3. Contribution of the Present Work

There are still some problems in the visual assessment using intracavitary contrast 2-D echocardiograms. The most frequent problem is the presence of a high level of background noise that either prevents the detection of the microbubbles or may simulate their appearance [5], which may bring in difficulties and incorrect evaluations of contrast 2-D echocardiograms for some patients. However, the microbubbles can still be detected by visual inspection according to their characteristic movement patterns and physician's
empirical intuition as well. Besides, physicians usually give visual assessment results by observing both long-axis and short-axis views of the contrast 2-D echocardiograms in many cardiac cycles, and also using multi-injection of contrast agent to avoid some possibly false assessment results caused by technical difficulties or by temporary physiological variations of patients. The highly correct MR assessment results can, therefore, be obtained by visual inspection in the contrast 2-D echocardiograms, but it takes considerable time, hard patience and highly skilled clinical experience, which only an expert cardiologist is able, and also only gives qualitative assessment results.

The work we developed resulted in the MIRas Regurgitation Assessment System (MIRAS) for contrast 2-dimensional (2-D) echocardiograms as a clinical diagnostic aid, to solve subjective, qualitative and time consuming problems of the visual assessment. MIRAS simulates the analysis process of a physician and mainly performs the following operations: sequential image acquisition and automatic alignment of the contrast 2-D echocardiograms; semiautomatic definition of regions of interest (ROIs) in the grabbed sequential images; adaptive detection of microbubbles from high level background noise; MR features extraction from microbubble distribution and motion parameters; contrast 2-D echocardiogram cases classification in 4 classes; generation of classification report; authorized upgrading of the classification database. User-friendliness has also been a point of concern enabling this system for practical use in a clinical environment.

The automated analysis and interpretation of the contrast 2-D echocardiograms performed by MIRAS is therefore an attractive solution having in view to overcome the mentioned insufficiencies. Results of 61.8% correct classification rate and 90.5% partially correct classification rate have been obtained for this system when using 202 poor quality image sequences obtained from the multi-injection echocardiograms of the 157 patients through the systematic management of image data and consistent processing with reasonably low degree of human intervention. Although the automated assessment results are a little inferior to the visual assessment ones, however, taking into account the advantages of the system, MIRAS is still a quite attractive medical aid to MR assessment.

MIRAS can be operated by a non-expert to assess the mitral regurgitation according to the hints in the software menus. The values of extracted features and classification results shown in an assessment report can help users describing and understanding the mitral regurgitation criteria in a quantitative way. The difference between calculated maximum and sub-maximum probabilities of classes can tell quantitatively the ambiguity
of the case mitral regurgitation between two classes. The speed of assessing an image case is reasonably high since it takes only about 20 minutes when using a 12MHz PC/AT personal computer system. The speed can easily be improved by using a PC-386 or a faster personal computer.

2. System Overview

The development of MIRAS comprised the following innovative contributions\(^{[23]}\)\(^{[27]}\) to the state-of-art of the problem:

- The development of sequential image acquisition, which is suitable for grabbing image sequences of a number of phases in any kind of cardiac images.

- The development of an efficient image alignment method overcoming misalignment of corresponding image elements in successive cardiac cycles. This misalignment problem is frequently encountered in sequential cardiac image analysis and has been recognized as one of the most difficult problems\(^{[28]}\)\(^{[30]}\).

- The study and development of an ROIs (regions of interest) definition method suitable for 2-D echo cavities in contrast-free image and subsequent contrast images.

- The development of algorithms for reliable adaptive discrimination of contrast microbubbles in the cardiac cavities from the high level background noise, based on linear regressive models.

- The derivation and study of MR features by analyzing microbubbles motion patterns based on their echo intensities and their distributions.

- The study and development of hierarchical linear classifiers for classification of the presence and severity of MR in four classes, yielding a 61.8% correct classification rate and a 90.5% partially correct classification rate.

These algorithms are sufficiently generic to be used in other applications, such as aortic valve regurgitation assessment, tricuspid valve regurgitation assessment, and myocardial perfusion image analysis.
2. System Overview

In order to be easily applied in hospitals and clinical centers, MIRAS was developed on a popular low cost personal computer system with simple peripheral devices, easily bought and installed. The software system was developed mainly in C language as well as in assembly language. The data processing is automatically managed by MIRAS internal structures. The automatic analysis process was designed to be performed with the less possible user intervention. Each process function can be chosen by using mouse or keyboard to select the appropriate menu option which is easy to learn and to access.

2.1. Hardware System Configuration

MIRAS was developed using a TOPIS 286 PC/AT compatible personal computer. This computer has a 80286 microprocessor, a 80287 math co-processor, a memory of 1 Mbytes, a 2-channel RS-232C interface, a printer interface, a PEGA card, a 1.2 Mbyte and a 360 Kbyte floppy diskette drivers, a hard disk of 40 Mbytes, a model 5560 keyboard and a three-button mouse (left and right buttons were used).

The PEGA card is fully compatible with IBM Enhanced Graphics Adapter hardware and software. It can support a series of display modes. The 80 column text mode with 8 colors and the 640x350 enhanced color graphics mode with 16 colors were used in the design of MIRAS.
2.1.1. Functional Structure and Main Characteristics

The MIRAS was based on the personal computer with DT2851-50Hz High Resolution Frame Grabber and DT2858 Auxiliary Frame Processor boards, as illustrated in Figure 5 [23]. Contrast 2-D echocardiograms are played from a VHS video recorder which is controlled by a remote control unit with several functions. These function commands can be given by finger touch or sent through an RS-232C serial interface and executed by a control adapter. Actually, the control adapter is not often used because it needs reasonable quality simultaneous ECG superimposed on the images which is usually hard to record. The text monitor of the personal computer was connected with a PEGA card that was plugged in a slot of the computer for demonstrating menus and data outputs. The RGB video monitor allowed the demonstrations of digital images. The digital images can also be printed by a video graphic printer which is an optional device. A final report can be printed out by a text printer.

A SANYO video cassette recorder (VCR), VHR-D500EX-G, based on PAL-VHS signal system is used for displaying recorded contrast 2-D echocardiograms. The video recorder is accompanied by a remote control unit which allows to perform the functions of STOP, PLAY, FAST FORWARD, REWIND, PAUSE, SLOW, FRAME ADVANCE, REPEAT, SEARCH, etc.
PAUSE, SLOW and FRAME ADVANCE are very important functions for image acquisition besides the normal ones of STOP, PLAY, FAST FORWARD and REWIND. The PAUSE function can freeze an image from playback state and let it last for about five minutes. FRAME ADVANCE function can advance the playback image frame by frame. SLOW function can advance the tape at approximately 1/10 slow-motion from playback or pause state. The slow-motion speeds can vary from 1/5 to 1/25 of normal speed and be changed to playback or pause state. The slow motion is useful for advancing the image from one frame to another in a proper speed and a controllable way. the PAUSE function will then freeze the image for subsequent data acquisition.

The video control adapter attached to the VCR remote control was specially developed in our Department for conveying computer generated control commands to the VCR control unit. The connection with the computer is made through an RS-232 serial interface. The remote control adapter through the VCR control unit can then generate the execution of the same video functions as the VCR control unit does. These functions can be executed by pressing keys on the keyboard or by calling subroutines in programs. Therefore this video adapter allows the automatic image motion control through a loop (as shown in Figure 6). However, the video control adapter is an optional device in MIRAS, because most of the time the manual control of image playback is performed and satisfactory.

![Diagram of video control adapter](image)

**Figure 6**. Automatic image motion control loop realized by the control adapter

### 2.1.2. Image Acquisition and Processing Boards

The image acquisition was performed through the DT2851-50Hz High Resolution Frame Grabber [31] which was installed in the PC/AT compatible computer. A
commercially available software package DT-IRIS is designed for the DT2851 frame grabber and the DT2858 auxiliary frame processor. Block diagrams of these boards are illustrated in Appendix. The main configuration of the frame grabber and its data exchanges with the peripheral devices as illustrated in Figure 7.

**Figure 7.** Main structure of the frame grabber and its data exchanges with the peripheral devices

The frame grabber has eight 256x8-bit RAM input look-up tables (LUTs), two 512x512x8-bit high speed frame-store memory buffers, eight 256x24-bit RAM output LUTs, and three 8-bit video RGB outputs (each for red, green and blue).

The two 512x512x8-bit frame-store memory buffers (256 Kbytes each) allow the digitized input signal to be stored in memory where it can then be accessed over the PC/AT bus at normal microcomputer bus speeds. These frame-store memory buffers are jumper selectable to occupy any of the 512 Kbyte blocks within the PC/AT memory space. In the 1Mbyte memory of the TOPIS PC/AT compatible personal computer, one extended frame buffer is available.

The DT2851-50Hz High Resolution Frame Grabber is supported by DT2858 Auxiliary Frame Processor [32] for high-speed 16-bit image processing. It greatly reduces the time required to accomplish arithmetic-intensive operations on 512x512 image frames.

The DT2858 Auxiliary Frame Processor performs addition and subtraction of a constant, multiplication by a constant, arbitrary non-linear transformation of pixel values, addition or subtraction of two frames, AND, OR or XOR of two frames, frame
averaging, convolutions of arbitrary shape and size, division and normalization, histogramming, zoom, pan and scroll.

2.2. Software System Configuration

The MIRAS software was basically developed in C language combined with some assembly language subroutines in Quick C programming environment. C language is a general-purpose programming language well known for its efficiency, economy, and portability. In many cases, well-written C programs are comparable in speed to assembly-language programs, and they offer the advantages of easier maintenance and greater readability.

C run-time library and C graphics library are especially important in C programming because C programmers rely on the library for basic functions not provided by the language. Most of these functions are important to the MIRAS development.

C library manager can help us to create, organize, and maintain run-time libraries so that the run-time routines we have developed can be managed in or called from a library.

Besides, C language routine modules can be mixed with assembly language routine modules, and the subroutines of DT-IRIS software package [33] can be used in C language program environment. All these advantages of the C language mentioned made it a good choice for the MIRAS programming language.

Another important tool for MIRAS software development was DT-IRIS [33]. DT-IRIS is a library of subroutines which support all the functions of the DT2851 frame grabber and DT2858 auxiliary frame processor. It is intended for use in a program development environment where application programs are written. DT-IRIS supports the following important categories of operations:

- Acquisition of video frames into the DT2851 frame grabber's frame-store memory buffers.
- Storage and retrieval of images to and from disk files.
- Display and control of a display cursor.
- Logical operations.
- Graphics overlays.
- Statistical operations commonly used in image contrast and spatial enhancement including histograms and averaging.

MIRAS software consists mainly of image processing and pattern recognition parts as shown in Figure 8. The purpose of the image processing part is to detect microbubbles in the cavities of the contrast 2-D echocardiographic images for subsequent quantitative analysis of MR. Automatic classification of MR is realized through the analysis of microbubble motion and MR features selection in the pattern recognition part.

In the following we give an overview of the MIRAS software by describing briefly its main operations.

![Diagram](image)

**Figure 8.** General diagram of the software system
2.2.1. Image Acquisition

The main operation in the acquisition of contrast 2-D echocardiograms from video cassettes is their digitization in a series of single-frame images. The DT2851-50Hz frame grabber can digitize an entire video frame to a 512x512x8 bit matrix and store it in a frame-store memory buffer. The 512x512 pixel image size and 8 bit grey levels are more than enough to the analysis precision of contrast 2-D echocardiographic images, and they may occupy a lot of disk space and take a long computation time. Actually, the entire frame of the contrast 2-D echocardiographic image is not interesting to us, but only a certain region containing the cardiac chambers LV and LA as well as the mitral valve can be covered by a 256x256 pixel square region. Meanwhile, the two LSBs of an 8 bit pixel preserved for red, green and blue color overlays are necessary in image processing and analysis. Therefore, only a 256x256x8 bit matrix from a frame is acquired so as to save disk storage and reduce computational time. The upper 6 bits are used for the image intensity code and the two LSBs can then be reserved as color graphic mask overlays for definition of ROIs and for observation of image spatial alignment. The grabbed image sequence is composed of end-systole and end-diastole frames of one heart beat contrast-free image and five consecutive heart beat contrast images starting from contrast injection.

In sequential image analysis issues, alignment of each image to a certain template is very important for computer comparative analysis. Misalignment in successive cardiac images, caused by heart movement, breathing chest motion or other artifacts during image formation process, has been recognized as one of the most difficult problems by several authors. In this work, the cardiac position in the images was properly aligned by deriving a binary mask from the contrast-free image which contains the characteristics of the myocardial structure and mitral valve [26] during the image acquisition process. Both ES and ED masks are derived from the contrast-free images and stored in disk. During the contrast image acquisition, the masks are alternatively loaded to memory frame buffer and superimposed on their relevant phase contrast images. The myocardial structure and the mitral valve in the mask can be automatically aligned to those in the contrast image by applying a fast image matching method.
2.2.2. Regions of Interest Definition

Significant regions of the LA and LV chambers containing representative microbubbles of the chambers are defined as regions of interest (ROIs) for detecting the presence and distribution of the contrast microbubbles.

The endocardial borders of LA and LV in ES image and of LA in ED image were approximately traced by thresholding the contrast-free images to generate masks, which are in the form of a mesh, covering ROIs and the other areas. These mesh-like masks allow us to observe both the masks and the covered images. Hand tracing under keyboard or mouse control easily isolates the ROIs from surrounding areas. After identified, the ROIs are automatically selected by object labeling (assigning distinctive mask intensities to the ROIs).

ROIs of each contrast image were obtained by the adjustment in a ring neighborhood around the ROI of contrast-free image in an automatic way.

The defined ROI masks are assigned distinctive colors and overlaid on the images. The images with overlaid ROI masks are saved under the same file names.

2.2.3. Microbubble Detection

High level of background noise is often the major problem for evaluating MR in contrast 2-D echocardiograms. It may either prevent the detection of the microbubbles or simulate their appearance to generate a false positive result.

Visual inspection of the contrast 2-D echocardiographic images shows that a contrast microbubble spot usually has a larger dimension [23], above a certain (high) intensity level, and higher intensity than a noise spot. In most images, a noise spot generally occupies approximately a 2x2 pixel area above the baseline intensity.

In the microbubble detection phase, noise spots in the ROIs of images are removed by using linear regressive models of microbubble detection, which were derived from a set of 73 randomly selected image sequences. The noise cleared contrast
echocardiographic images can be used both for quantitative analysis and for clinical visualization.

2.2.4. Microbubble Motion Parameter Extraction

According to the MR criteria, the image analysis is based on evaluating the amount of microbubbles moved from LV to LA through an eventual mitral valve defect, and their distributions in LA. In this phase, seven basic parameters of the contrast images are measured, including ROIs areas in LV and LA, covered areas of microbubbles in LV and LA, sums of echo intensities of microbubbles in LV and LA, standard deviation of microbubbles spatial distribution in LA. In order to form the measurement space of MR pattern recognition, sixty eight descriptors of microbubble motion patterns were extracted based on the seven basic parameters characterizing the LV and LA ROIs in the ES and ED phases, for five heart beat contrast images [27].

2.2.5. MR Feature Selection

The sixty eight descriptors of microbubble motion patterns may be considered as the MR measurements. With the help of SPSS statistical package [34], some features were tested and selected from the measurement space by using 202 class labeled image sequences as a training set with F statistic and stepwise selection method. Based on the Bayes maximum likelihood rule, some potentially useful hierarchical linear classification trees were built. A classification tree structure was selected according to the minimum total misclassification probability which consisted of three successive dichotomies. The clinical significance of the selected feature space was interpreted.

2.2.6. Linear Classification

Based on the training set database of the 202 class labeled image sequences, the hierarchical linear classifier was designed, with the selected features. For the classifier performance estimate, resubstitution method and rotation method [35] were employed to test the classifier performance in the 202 image sequences. The resubstitution method uses the whole training set as the test set to obtain the upper bound of the true
performance. The lower bound can be obtained by dividing the 202 image sequences into three test sets and using the rotation method. Therefore, we obtained the upper bound estimate from the whole training set as 64.4% correct classification rate. Considering the clinical reality of the relaxed case labeling, the upper bound of the partially correct classification rate was obtained as 91.7%. For the lower bounds of those two rates we respectively obtained 59.2% and 89.3%. Therefore, the true performance of the classifier can be estimated as 61.8% for correct classification and 90.5% for partially correct classification. The classification results can be automatically reported and viewed by clinicians. The database for classification can be updated with new classification data.

3.1. Sequential Image Acquisition and Alignment

As in other modes of cardiac image processing, and in image processing in general, the process of digitization can be described as dividing an image into a grid or matrix of discrete pixels, each pixel located by its X and Y coordinates. At each pixel position on image attribute is measured and assigned a discrete value. In ultrasound images, the attribute of interest is local echo amplitude, displayed as grey level. Thus the digitized contrast 2-D echocardiograms may be described as a matrix of pixels, each pixel having a discrete grey level corresponding to an echo amplitude value.

This image digitization process can be performed by the commercially available DT2851-50Hz frame grabber, as described in Chapter 2, which has a 512x342x8 bit matrix. The contrast 2-D echocardiograms from video cassette were periodically digitized in a series of single-frame images. Only 256x256 pixel square region of 6 bit
3. **Contrast Echocardiogram Processing**

Contrast 2-D echocardiograms recorded from an echocardiograph is a series of continuous real-time images which usually have a poor quality and interference from various noises. They often provide superfluous information about mitral regurgitation. Meanwhile, the scattered noise spots in the contrast images often bring about some difficulties for visualization of microbubbles, not to say for automatic quantitative assessment. Proper number of images acquisition for the assessment of mitral regurgitation and improvement of the images quality are important for assessment feasibility and accuracy. In this chapter, the preparation of contrast echocardiographic images and the processing of these images for quantitative assessment of mitral regurgitation are described.

3.1. **Sequential Image Acquisition and Alignment**

3.1.1. **Number of Image Frames**

As in other modes of cardiac image processing, and in image processing in general, the process of digitization can be described as dividing an image into a grid or matrix of discrete pixels, each pixel located by its X and Y coordinates. At each pixel position an image attribute is measured and assigned a discrete value. In ultrasound image, the attribute of interest is local echo amplitude, displayed as grey level. Thus the digitized contrast 2-D echocardiograms may be described as a matrix of pixels, each pixel having a discrete grey level corresponding to an echo amplitude value.

This image digitization process can be performed by the commercially available DT2851-50Hz frame grabber, as described in Chapter 2, which has a 512x512x8 bit matrix. The contrast 2-D echocardiograms from video cassettes were periodically digitized to a series of single-frame images. Only 256x256 pixel square region of 6 bit
grey level were grabbed for disk space saving and image spatial alignment, as well as for graphic overlay. With this frame grabber and the following procedures, 202 image sequences were grabbed from the 157 patient contrast 2-D echocardiograms.

The software has been developed to be flexible enough for a general periodic image acquisition. It is able to grab one calibration image cycle and up to 25 subsequent image cycles. In each periodic cycle, up to 36 image phases can be grabbed. The general purpose image acquisition software can thus perform calibration image acquisition, subsequent image alignment and acquisition, as illustrated in Figure 13.

3.1.1. Image Acquisition

Since the original contrast 2-D echocardiograms contain too much information about mitral regurgitation, selection of some typical image samples from the echocardiogram is important for mitral regurgitation information extraction and for system realization. If too many image frames are selected for analysis, the advantages of an assessment system based on a personal computer may not be obvious because of the memory space and computational time involved. On the other hand, insufficient images do not provide enough mitral regurgitation information yielding poor assessment results.

3.1.1.1. Number of Image Frames

By studying the properties of the contrast 2-D echocardiograms and the established criteria for mitral regurgitation, it is found that most mitral regurgitation information is present in the first heart beats of the contrast 2-D echocardiograms starting from the first appearance of the contrast microbubbles. As a compromise between mitral regurgitation information and computer resources, one cardiac cycle of contrast-free image before contrast agent injection and five cardiac cycle of contrast images after contrast agent injection are captured. The feature analysis and classification results described thereafter will show that the five heart beat images are enough for a sufficient MR assessment.

Although the mitral regurgitation usually happens during the cardiac systole which provides therefore with most of the information, the contrast microbubbles are not always observable in positive mitral regurgitation case, which is the so called false-negative,
during the cardiac systole. However, the regurgitated microbubbles then circulate in the LA chamber and may appear in the image observation plane during the cardiac diastole. The cardiac diastole therefore also provides additional information for compensating the possible false negative presence happening during systole.

For instance, contrast microbubbles in the images of a mild positive mitral regurgitation case may not be visualized because the jet effect originated by small orifices in the mitral valve corresponds to jetting contrast microbubbles out of the heart optical observation plane. These off-image contrast microbubbles randomly circulating in the LA cardiac chamber might appear in the next diastole of the cardiac cycle. For this reason, both end-systole and end-diastole image phases of the contrast-free image and the contrast images are acquired. Examples of grabbed 256x256 portion images of end-systole and end-diastole phases both in contrast-free and contrast images are shown in Figures 9, 10, 11, 12.

Figure 9. A significant region of a 256x256 pixel matrix in a ES frame of a contrast-free image
Figure 10. A significant region of a 256x256 pixel matrix in a ED frame of a contrast-free image.

3.1.1.2. Image Files

For acoustic processing, the required sequenced image frames.
the image file name is a sequence of characters, usually the concatenation of four.
the patient name and the two characters in the 8-character
are reserved for the frame number.
the last two characters in the file name indicate the ES or ED image phase, respectively.
The last two characters identify the heart cycle image and
the first is the ES or ED cardiac cycle image.
frames extensions are always.
the letters of image file names are therefore.

Figure 11. A significant region of a 256x256 pixel matrix in a ES frame of a contrast image.
3.1.1.2. Image Files

For automatic processing and management of the acquired sequential image frames, the image file names are arranged in an indexed way. In a file name up to eight characters, the first 6 or less characters are used for case identification, usually the abbreviation of the patient name with a serial number discriminating from others. The last two characters in the 8-character file name are reserved for the indexes of the sequential image frames. The last but one character in the file name is "A" or "B" for identifying the ES or ED image phases, respectively. The last character in the file name is for identifying heart beat number with 0, 1, 2, 3, 4, and 5 representing the contrast-free cardiac cycle image and the first to the fifth contrast cardiac cycle images, respectively. The file names extensions are always "IMG". Examples of image file names are therefore as follows:

A1A0.IMG, AB001B1.IMG, A00123B3.IMG, ABCDE1A4.IMG
In the image file names above, the underlined characters are the abbreviations of patients names and their numbers, the bold letters A and B denote ES and ED phases, respectively; the bold digits 1, 3 and 4 denote the cardiac cycle numbers of the contrast images, and the bold digit 0 denotes the contrast-free image before contrast injection.

Actually, the system was designed flexible enough for other periodic image acquisition purposes. The last but one character of the file name can be one of the 26 alphabet letters from A to Z, and the last character can be one of the ten numbers from 0 to 9 or one of the the 26 alphabet letters from A to Z. Therefore it is able to acquire the maximum 36 period images and the maximum 26 image frames in each period. The order of the image period number is 0, 1, 2, ..., 9, A, B, ..., Y, Z. The period number and frame number of each period can easily be set by initial system options of MIRAS as shown in Figure 13. The system can automatically access the image files for image processing by indexing their file names. The contrast echocardiographic images of a large number of patients can therefore be easily managed. The flowchart of the contrast echocardiographic image acquisition procedure is shown in Figure 13 on the following page.

3.1.2. Temporal Alignment of Image Frames

In order to grab different heart beat but superimposable images in the same cardiac phase (end-systole or end-diastole), temporal alignment must be performed either by manual control of the video cassette playback or by automatic control of the video through the video control loop as described in Figure 6. The automatic control needs a simultaneous ECG wave superimposed on the contrast 2-D echocardiograms.

3.1.2.1. Manual Control Alignment

The usual way to perform temporal cardiac frame alignment is by manual control of the video playback while watching the image display on the screen. The image cardiac cycle can be aligned by stopping and grabbing an image when it is in the same cardiac phase as other images of the same image sequences.
System initial options:
i can be assigned up to 35 image cycles
j can be assigned up to 26 image phases

Set image phase j = 'ES'

Display of the j phase of the contrast-free image

Derivation of a binary template from the image

Saving 256x256 pixel portions of both template and image

j = 'ES'? Y Set j = 'ED'

N

Loading both ES and ED 256x256 pixel templates to two parts of a frame buffer

i = 1 j = 'ES'

Display of the j phase template superimposed on contrast images

Adjusting the j phase of the ith heart beat contrast image to the present template for a temporal alignment

Derivation of a binary image from the j phase of the ith heart beat contrast image

Adjusting the superimposed template to the contrast image by the binary image matching for a spatial alignment

Saving the 256x256 pixel contrast image portion covered by the template

j = 'ES'? Y Set j = 'ED'

i = i + 1

N

i = 5? Y STOP

Figure 13. Flowchart of the image acquisition procedures for the contrast 2-D echocardiograms.
MIRAS for Contrast 2-D Echocardiograms  Echocardiogram Processing

In the MIRAS system, the ES and ED image phases of the contrast-free images are successively displayed, stopped and grabbed. 256x256 pixel portions of the contrast-free images, which contain the main characteristics of the myocardial structure and the mitral valve [25], are grabbed and stored in disk. Binary templates of the 256x256 pixel portions are generated from the the contrast-free images by thresholding, and stored in the active disk with special file-names "A.CLB" and "B.CLB", respectively, as shown in Figures 14, 15 and 16, for guiding the subsequent contrast image cardiac phases alignment, as shown in Figures 17 and 18.

The templates are alternatively loaded from the disk to memory frame buffer and superimposed on the contrast images for temporal alignment. In the same way, all the relevant contrast image phases can eventually be aligned to each other. After each temporal alignment is performed, a spatial alignment, as described thereafter, will be performed for the contrast image with the same template. Alternatively, the next spatial alignment will also be performed following the next temporal alignment. In this way, temporal alignment guided by the calibration masks is obtained with much more convenience and better quality than by simple visual inspections.

Figure 14. Selected 256x256 pixel region in an ES phase of contrast-free image
Figure 15. Threshold level finding in the histogram of the contrast-free image in the square region in Figure 14 and its mapping to binary template

Figure 16. Template derived from the contrast-free image of Figure 14 after thresholding
Automatic Control Alignment

In the design of the control loop described in Chapter 3.1.1.1, the triggering of the superimposed ECG signal is provided by an adaptor and the VCR control. An R-wave is used for moving in a control loop.

Figure 17. Temporal alignment of cardiac phase compared with template by manual control before match point

Figure 18. Temporal alignment of cardiac phase compared with template by manual control at match point
3.1.2.2. Automatic Control Alignment

The automatic control of video playback is performed through the control loop described in Chapter 2. The control commands are sent by R-wave triggering of the superimposed ECG. The commands are converted to control signals by the VCR control adaptor and the remote control unit to stop video play when an expected R-wave is moving to a certain coordinate of the screen.

In the development of the automatic control method, there were available only two cases of contrast echocardiograms with superimposed ECG. The ECG wave starts from the vertex of the image sector and horizontally goes to the left border of the screen. The ECG wave image on the screen is always blurred and has a certain thickness. Its intensity is also inhomogeneous. For obtaining the ECG pattern from the blurred ECG wave image, ECG thinning is made by tracing the upper boundary of the wave. Thinned ECG is shown in Figure 19. An R-wave detection algorithm is applied to the traced ECG wave.

![Figure 19](image_url)

Figure 19. The thinned ECG (left) and the original blurred ECG image (right).
There are always small noise spots in the dark background whose echo intensities are usually below 30. For avoiding the noise interference, pixels detected below intensity 30 are considered to be the background. Therefore, only the pixels above intensity 30 in the ECG image are detected for the ECG thinning and the characteristics deriving. The algorithm for detecting R-wave requires positive R-wave because the superimposition of negative ECG wave on the image sector will cause difficulty for recognizing the ECG image.

In the automatic cardiac frame alignment and images acquisition, the ES and ED phases of contrast-free images are, as before, manually displayed from the video and stopped for deriving characteristics of the R-wave including its coordinates. The search for R-wave starts from the origin of the ECG to the left border of the screen. When the first maximum Y coordinate, corresponding to the R-wave, is found in ES and ED contrast-free image phases, the X coordinates are recorded characteristics for the ES and ED image phases, respectively. Meanwhile, the difference between the maximum Y coordinate and the average of the Y values at 2 pixel steps before and after the present X coordinate is measured as the characteristic of the detected R-wave.

In the same way, the maximum Y coordinate and the difference of the Y coordinates are real-time measured for searching the subsequent R-waves pertaining to the contrast images. When the difference value is measured larger than 0.4 times the absolute value of the Y difference characteristic in ES or ED contrast-free image phase, and the X coordinate drops in the neighborhood ranging from -3 to 3 of X coordinate characteristic of the ES or ED contrast-free image phase, the R-wave is considered to have been detected for the ES or ED contrast image phase, respectively. Therefore, the corresponding ES and ED contrast image phases are obtained for thereafter spatial alignments. This R-wave detection process is illustrated in Figure 20.

3.1.3. Spatial Alignment of Cardiac Chambers

Spatial misalignment in cardiac images, caused by heart movement, breathing chest motion or other artifacts during the image formation process, is always encountered in sequential cardiac image analysis applications. It has been recognized as one of the most difficult problems by several authors[28]-[30]. In other cardiac image analysis systems
that have been reported, the image sequences are usually captured and compared with each other to make sure that they are aligned before processing. Any sequences in which one or more of the frames has moved in relation to the others are discarded. [30], amounting to an important loss of data. Because of the absence of a good alignment, no

R-wave characterization at ES and ED:

\[ Y_{\text{dif}} = \frac{Y_R - (Y_F + Y_B)}{2} \]

Characteristics:

- \( X_{\text{ES}} \) or \( X_{\text{ED}} \) corresponding to maximum \( Y \)

R-wave detection at ES and ED:

When the maximum \( Y \) is detected in the region \([X_{\text{ESF}}, X_{\text{ESB}}]\) or \([X_{\text{EDF}}, X_{\text{EDB}}]\), and the \( Y \) difference \( > 0.4 \times \text{abs}(Y_{\text{dif}}) \), the R-wave is detected at ES or ED.

Figure 20. Illustration of R-wave characteristics in ES and ED contrast-free image phases and the detection of R-waves in ES and ED contrast image phases
motion artefact or breathing can be then tolerated during data acquisition. In the present system, the misalignment problem is basically solved with a developed image alignment algorithm, based on an image matching algorithm, in the captured square portion of cardiac images. The developed algorithm works well even in the presence of motion artifacts or breathing as long as there is no rotation motion or cross-sectional movement relative to the image plane.

3.1.3.1. Image Matching Algorithm

When we match a template against an image, e.g., by cross correlation, we may expect to find any exact copies of the template eventually present in the image. These copies must be of the same size, however, and have the same orientation, as the template; we will not be able to find a rotated or enlarged version of the template in the image, since such a version will not, in general, give rise to high values of the cross correlation. Similarly, if a geometrically distorted copy of the template is present, we will have difficulty detecting it by matching unless the distortion is quite small.

If we want to use template matching to recognize a pattern that is subject to rotation, scale change, or other geometrical transformations, a very large number of templates would ordinarily be needed. Another possibility, however, is to use a single template, and search through the space of permissible distortions or transformations of the template, in an attempt to optimize its degree of match with the image. However, it is not practical to match the rotation, scale change, and other geometrical transformations with MIRAS because of the computation capability of the personal computer.

When we measure a degree of match in a given position between a template $h$ and an image $g$, we would like to be able to reject mismatches rapidly, since most positions will be mismatches. For this purpose, it is convenient to use a sum of absolute differences measure of mismatch $\Sigma |g-h|$. Given a mismatch threshold $t$, we want to compute $\Sigma |g-h|$ in such a way that, in a mismatch position, the sum can be expected to rise above $t$ as rapidly as possible. One way to do this is to measure $|g-h|$ first for those points of $h$ whose gray levels have high expected absolute differences from the gray level of a randomly selected point of $g$. The expected contributions of such points to $\Sigma |g-h|$, when we are not in a match position, are large; hence when we measure $|g-h|$ for those
points first, the sum $\sum |g-h|$ should tend to rise more rapidly than if we measured $|g-h|$ for the points of $h$ in some arbitrary order.

3.1.3.2. Automatic Spatial Alignment

In this system, the myocardium and mitral valve positions in the image sequences are properly aligned to each other by using the previously derived 256x256 pixel binary templates of the contrast-free images, shown in Figure 21, during temporal image alignment process. After the contrast image temporal alignment, the templates which have been alternatively loaded to the memory frame buffer are superimposed on their relevant phases of contrast images, as shown in Figure 22. The binary templates can be shifted vertically or horizontally in a 512x512 pixel contrast image area because they are of 1/4 size of the images. The myocardial structure and mitral valve in the template can be automatically aligned to those in the contrast image after applying a rapid matching method [26] to be derived next, as shown in Figure 23. The 256x256 pixel portion of the contrast image under the template is grabbed and stored in disk. In this way, all the images in the same image sequence are aligned to each other by the myocardium and mitral valve coordinates.

Figure 21. A 256x256 pixel binary template derived from a contrast-free image in ES phase.
3.1.3.3. Image Matching Realization

Because convolutions and calculations should be matched in the image which is the best way to match the data (especially in personal computers). To realize this we noticed that when the myocardial movement range of each subject is fixed and the data of the myocardial movement and its spatial relationship to myocardial structure in an image can be used to reduce the image matching area to the 256x256 pixel region covering the mitral valve.

After obtaining the image we noticed that the myocardial movement range of each subject can be aligned to each other as long as the mitral valve positions are aligned. Meanwhile the mitral valve in and out of ED phase is consistent. The position of the mitral valve and its spatial relationship to myocardial structure in an image results in the possibility to reduce the image matching area to the 128x128 pixel region covering the mitral valve.

Figure 22. The binary template superimposed on a contrast image in ES phase before spatial alignment

When a contrast image is located on a contrast image in ES phase image, the spatial relationship of the binary template of the 128x128 regions of ES phase in the ED phase in the matching operation is employed for the matching operation.

Figure 23. The binary template superimposed on the contrast image in Figure 22 after spatial alignment
3.1.3.3. Image Matching Realization

Because the image matching algorithm searches some reject points and calculates
convolution or the sum of differences between the template and the would-be matched
image which takes much computational time and a lot memory space (especially in a
personal computer) even with the rapid mismatches reject method, it would be unpractical
to realize the image spatial alignment in the system without reducing the matching area
and the data.

3.1.3.3.1. Matching Region Definition

After observing and experimenting with the image spatial matching, we noticed that
the myocardial structures in the same cardiac phase of an image sequence can be aligned
to each other as long as the mitral valve positions are aligned. Meanwhile, the mitral valve
in an image is usually limited to a 128x128 pixel region. These characteristics of the
mitral valve and its spatial relationship to myocardial structure in an image make it
possible to reduce the image matching area to the 128x128 pixel region covering the
mitral valve.

When a 256x256 pixel square region covering significant part of an image is located
on a contrast-free image, the mitral valve position is usually in the center position of an
ES phase image and in the upper center position of an ED phase image. The square
regions of 128x128 pixels in the center position of the 256x256 pixel square template of
the ES phase and in the upper center position of the 256x256 pixel square template of the
ED phase in the contrast-free image, as shown in Figures 24 and 25, are employed for the
matching operation.
Figure 24. The 128x128 pixel region (bright) covering mitral valve in the center position of 256x256 pixel binary template of the grabbed ES phase contrast-free image

Figure 25. The 128x128 pixel region (bright) covering mitral valve in the upper center position of 256x256 pixel binary template of the grabbed ED phase contrast-free image
3.1.3.3.2. Binary Images Operation

For applying the image matching algorithm in a fast and efficient way, a binary image is also derived from the contrast image using the same threshold as in the contrast-free image, as shown in Figure 26. In this way, the myocardial structure and the mitral valve in the binary contrast image is most similar to that in the binary contrast-free image template. Image alignment is actually performed between the 128x128 mitral valve regions of these two binary images (the binary contrast image and the binary template), and an exclusive-or logical operation and a sum calculation in the searching area. Although the microbubbles spots in the binary contrast image may simulate the myocardium or mitral valve, they are still less dense than the myocardium and mitral valve, especially in LA. The search for the matched position goes from up to down and from left to right. When the minimum intensities sum in the searching area is encountered, this position is considered to be the matched position.

![Figure 26. Binary contrast image in ES phase. The 128x128 pixel region (bright) covers the mitral valve](image-url)
In order to avoid using a large array to occupy the memory of a personal computer, each pixel data of the binary image is compressed to one bit in four LSBs of a character variable so that each character variable can contain 4 pixel binary image data (0 or 1). In the first search for matched position, each searching step is 8 pixels in the template present position neighborhood in which 9 positions are searched. When a matched position is found, a 4 pixel searching step is taken in the potentially matched position neighborhood in which 8 positions are searched to find more precise coordinates around the position, as shown in Figure 27. In this way, the last matched position is found which results in a 2 pixel matching precision or error, as shown in Figure 28, the matching precision satisfies completely the image alignment needs for which it takes only about 6 seconds to finish an image alignment.

**Figure 27.** The match position searching in 8 pixel and 4 pixel steps in the searching neighborhoods. The maximum matching error is 2 pixels. The circles in the figure denotes pixels.
3.2. Regions of Interest Definition

Mitral regurgitation assessment is based on the analysis of microbubbles flow characteristics from LV chamber to LA chamber. Both LV and LA chambers are defined as ROIs in the ES phase images, and only LA chambers are defined as ROIs in ED phase images because of the too small area of LV in the ED phase images. These chambers are defined as the regions of interest (ROIs). In the contrast images, the microbubbles in the chambers have high echo intensities comparable to that of the myocardial tissues, so that it is more difficult to define endocardial wall in the contrast images than in the contrast-free images. ROIs definition in the contrast-free images may be quite helpful for the definition of ROIs in the contrast images, since the myocardial structures in the contrast-
free image and the contrast images of the same image sequence have been aligned to the same coordinates during the image acquisition process.

In the MIRAS, ROIs in the contrast-free image are defined in a semiautomatic way. The ROIs in the subsequent contrast images are defined through automatic modification of the ROIs boundaries of the contrast-free image for adjusting the possibly incomplete spatial alignment and small variations of chambers walls. During the automatic definition process, the 256x256 pixel images are reduced to 128x128 pixels by replacing each disjoint four pixel set by their intensity mean, therefore accelerating the processing tasks. The ROIs definition procedure illustrated in Figure 29.

3.2.1. ROIs Definition in Contrast-Free Images

Although edge detection methods for 2-D echocardiographic images have been developed [37] - [45], in this application it is not necessary to accurately trace the endocardial wall using one of these time consuming methods. Approximate tracing of the endocardial borders of LV and LA chambers in contrast-free images can simply be done by performing a certain level of thresholding operation because the echo intensities of blood in the cardiac chambers are much lower than those of myocardial tissues. After thresholding is performed, which is derived from the histogram as in Figure 30, the darker regions in the image constitute a mask containing ROIs and other areas. In order to visually compare the image and its derived mask, two identical mask are formed. one mask is designed in the form of mesh, therefore allowing the observation of the covered image as shown in Figure 31. The other mask is a solid one stored in another frame buffer. Modification on the mesh mask is linearly mapped to the solid mask.

Since the raw mask originated from the contrast-free image by thresholding may have sharp boundaries, small holes in the ROIs and isolated small masks, as shown in Figure 32, it does not present the actual cardiac chamber shape and may cause some problems in the object labeling process described next. Therefore, it is necessary to smooth the mask boundary, fill up some small holes in the ROI mask and eliminate the small object pixels apart from the ROI mask. For these purposes, erosion and dilation operations are performed on the raw mask.
Figure 29. Flowchart of ROIs definition
Figure 30. Threshold derived from the histogram of the image in Figure 31

Figure 31. Mesh mask covering the darker regions in the contrast-free image of ES phase
In this system, erosion and dilation algorithms were developed to operate on binary images in which the higher echo intensity pixels have the value 1. The erosion and dilation are realized by convolution between a binary image and a template. Erosion or dilation operation depends on the template matrix value. In MIRAS, 5x5 and 7x7 erosion, and 5x5 dilation algorithms that we developed are shown in Figure 33. For processing the raw mask from the contrast-free image, a 5x5 dilation and a 5x5 erosion operations are performed. Smooth boundary masks covering the major portion of chambers are obtained as in Figure 34.
### 5x5 Erosion

<table>
<thead>
<tr>
<th>Binary image pixels</th>
<th>Template (X=1/20)</th>
<th>Integer kernel pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/0/0/1/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/1/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
</tbody>
</table>

When the number of bright pixels in the neighborhood of a kernel pixel is larger than 19, the kernel pixel is assigned to "1" otherwise to "0" ignoring its original value.

### 5x5 Dilation

<table>
<thead>
<tr>
<th>Binary image pixels</th>
<th>Template (X=1/6)</th>
<th>Integer kernel pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/0/0/1/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/1/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
</tbody>
</table>

When the number of bright pixels in the neighborhood of a kernel pixel is larger than 5, the kernel pixel is assigned to "1" otherwise to "0" ignoring its original value. (The kernel pixel is assigned to "1" when its result value is larger than 1)

### 7x7 Erosion

<table>
<thead>
<tr>
<th>Binary image pixels</th>
<th>Template (X=1/40)</th>
<th>Integer kernel pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/0/0/1/0/0/0/1</td>
<td>X X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/1/0/0/0/0/1</td>
<td>X X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
<tr>
<td>0/0/0/0/0/0/0/1</td>
<td>X X X X X X</td>
<td>0/0/0/0/0/0/0/0</td>
</tr>
</tbody>
</table>

When the number of bright pixels in the neighborhood of a kernel pixel is larger than 39, the kernel pixel is assigned to "1" otherwise to "0" ignoring its original value.

Figure 33. Illustration of 5x5 and 7x7 erosion, and 5x5 dilation algorithms

. 50 .
By using several mask modification tools that we have designed, consisting of graph-cut, polygon region definition, circle region definition, and rectangular region definition, the ROIs are easily isolated from other adjacent areas. It is usually sufficient that the user traces 1 or 2 lines limiting (cutting graph) the mask extent for obtaining quickly a small mask containing the relevant portion of the respective chamber, as illustrated in Figure 35 (LV ROI is separated from b and c segments). When the mesh masks are being modified, the same modification is also performed on the solid masks. However, the isolated ROI masks LV and LA are only separated from mask segments (a, b, c and d) outside the cardiac chamber in geometrical characteristics, as shown in Figure 35, but they have the same echo intensity which makes it difficult to distinguish automatically the ROI mask from the rest. For this reason, each isolated mask segment is then automatically assigned to a distinctive mask intensity by performing an object labeling operation.
The object labeling algorithm was developed for distinguishing the separated object pixels on a black (zero echo intensity) background. The algorithm assigns the same echo intensity to the joint pixels but different intensity from the disjoint pixels in ordinal numbers. Because of the 256 gray level of the image acquisition board, only 255 disconnected objects lying on the black background can be assigned to different echo intensities. The number of echo intensities is sufficient for labeling the isolated objects in this problem because some individual small mask pixels are eliminated and the number of isolated masks (objects) is quite limited. Besides, taking into account the mask modification with the manual tools, the diagonally joint pixels are considered to be separated pixels as shown in Figure 36.
Figure 36. Illustration of the object labeling algorithm. Each square denotes a pixel. In the labeled figure, each distinctive pattern denotes different intensity.

By using the object labeling algorithm, the isolated ROI masks and the isolated mask segments are assigned to different intensities as illustrated in Figure 37.

Figure 37. Distinctive mask intensity assigned to each isolated mask by object labeling algorithm.
After being identified by a mouse cursor controlled by hand, the ROI is automatically selected by the ROI mask echo intensity value as shown in Figure 38. The obtained ROI masks in the form of mesh and the respective contrast-free images are shown in Figure 39.

Figure 38. Selected ROI masks

Figure 39. ROIs mesh masks with covered contrast-free image of ES phase
3.2.2. ROIs Adjustment in the Contrast Images

Although the subsequent contrast images have been aligned to the contrast-free images, there are still some misalignment remaining between these images. There are also some variations between the endocardial walls of the several images. However, misalignments and shape variations usually happen in a near neighborhood of the contrast-free ROI border. By performing a 7x7 erosion and frame subtraction operations, a ring mask is obtained for modification of the contrast images, as shown in Figure 40. Higher echo intensity pixels in the ring region are considered to be the myocardium or mitral valve, and the lower echo intensity pixels are the cavities. The ROIs of the contrast images are adjusted by thresholding in the ring regions.

![Figure 40. A ring mask of an ROI border neighborhood in the reduced image](image)

Sometimes the bright pixels in the ring regions are microbubble pixels which are very near the endocardial wall. These microbubble pixels may be misrecognized as
myocardial tissues and therefore may be removed from the ROI ring regions resulting in new holes or defects in the ROI mask as shown in Figure 41, but the MR assessment results are not too much influenced by this misrecognition because the ring region is only a small ROI portion and meanwhile the new holes in the ROI mask can be filled by a sequence of operations.

Figure 41. Holes and defects in the ROI mask caused by the misrecognition of microbubbles as myocardium

In order to compensate the ROI masks for the misrecognition, the ROI mask is subtracted from a large binary mask (the 256x256 pixel mask is enough) of the same intensity as the ROI, as illustrated in Figure 42. By performing the object labeling algorithm, the subtracted masks are assigned to different echo intensities. Removing the largest mask pixels, the remaining small masks are the holes pixels in the ROI. By joining the holes masks to the newly derived ROI mask, the holes in the ROI mask are filled. Therefore the holes caused by misrecognition of microbubbles as myocardial tissues are compensated by the additional operations, as shown in Figure 43.
Figure 42. Process of filling holes in the ROI mask. Operation 1: subtracting the original ROI mask from a 256x256 square mask of the same intensity; Operation 2: labeling the subtracted masks; Operation 3: removing the largest mask resulting in the holes masks; Operation 4: Adding the holes masks to the original ROI mask.

As a matter of fact, the misrecognition mostly happens when microbubbles completely fill a cardiac chamber, usually the LV, in which the microbubbles often distribute homogeneously. Thus, the ROI area is simply reduced when microbubble pixels are misrecognized as myocardial tissues and this does not influence the MR assessment results. When the ROIs are adjusted to the contrast images, they are assigned to red color for LA and green color for LV, and superimposed on the respective contrast images chambers to be stored in disk under the same file names as the original images, as shown in Figures 43 and 44. In this way, the ROI masks can be easily separated from the images when the ROI images are analyzed.
3.3. Detection of Microbubbles

In the echocardiographic image processing issue, strong background noise is the most important issue that has been recognized as one of the main problems in contrast echocardiography assessment. To handle background noise because there is a need to identify the microbubbles for their similar characteristics and then evaluate their characteristics. Although there is relatively stable movement in the moving microbubbles, identifying the microbubbles from the moving spots in the image is not possible. The microbubbles' movement pattern is also similar to the image obtained from the moving cardiac chambers. The motion of the microbubbles may simulate the microbubbles' appearance in the frozen image frames.

Figure 43. ES phase contrast image with adjusted ROIs mask overlay

In the MR assessment problem, we used the 202 independent image sequences grabbed from the 157 patient echocardiograms overcoming the limited number of cases to be analyzed. Thus, a considerable number of image sequences were taken from the multiple injection images which were considered to be unsatisfactory for visual assessment with one sequence alone because of their poor quality. It is more important and more difficult to improve microbubbles detection from the noisy image sequences. The principal task in the image processing procedure is illustrated in Figure 44.

Figure 44. ED phase contrast image with adjusted ROI mask overlay
3.3. Detection of Microbubbles

In the echocardiographic image processing issue, strong background noise is the most important factor reducing the image quality, and its elimination has been recognized as one of the most difficult problems. Moreover, the MR assessment in contrast echocardiographic image is more sensitive to the influence of background noise because the noise spots are easily confused with the contrast microbubbles for their similar characteristics. These noise spots may either prevent the detection of the microbubbles or simulate their appearance so as to cause false-negative or false-positive results. Although physicians can usually distinguish the moving microbubbles from relatively stable noise spots in the playback of contrast echocardiograms by observing the microbubbles motion patterns, it is, however, impossible in the MIRAS system to recognize the microbubbles movements because both microbubbles and noise spots are standing still in the grabbed image frames. Besides of the difficulty of the microbubbles recognition from the noise spots, the background inhomogeneity of the echo intensities in the cardiac chambers may also simulate the microbubbles appearance in the frozen image frames.

In the MR assessment problem, we used the 202 independent image sequences grabbed from the 157 patient echocardiograms overcoming the limited number of cases to be analyzed. Thus, a considerable number of image sequences were taken from the multi-injection images which were considered to be unsatisfactory for visual assessment with one sequence alone because of their poor quality. It is more important and more difficult to improve the image quality in this problem. Therefore, microbubbles detection from the noisy inhomogeneous background of the cardiac cavities is the principal task in the image processing section. A general view of microbubbles detection procedure is illustrated in Figure 45.

3.3.1. Microbubbles Detection Based on their Characteristics

Visual inspection and experimental studies have shown that the microbubbles spots above a certain (high) intensity level are always geometrically larger than the noise spots. Therefore one of the approaches to the detection of microbubbles is measuring the maximum size of noise spots in the cavity of the contrast-free image and eliminating the
spots in the contrast images whose sizes are lower than the measured maximum size [23]. The contrast microbubbles can basically be obtained in this way.

Figure 45. Flowchart of microbubbles detection
Besides, it has also been found that the intensities of noise spots are usually lower than those of the microbubbles spots, and each noise spot generally occupies approximately 2x2 pixel area above the baseline intensity. The noise smoothed baseline of the cavity background can then be obtained by performing a 2x2 lowpass filtering on the contrast-free image. By digitally subtracting the noise smoothed baseline image from the original contrast-free image, we obtain the noise spots image with the intrinsic noise spots intensity values in the background normalized image. The maximum intensity value $N$ is measured in the ROI of the noise spots image for separating the noise spots from microbubbles spots. In order to reduce the effect of the inhomogeneous background intensities and to eliminate the noise pixels from the contrast image, the contrast image is also normalized by subtracting the noise smoothed contrast-free image from the contrast one. In these images subtractions, any resulting negative values were set to zero. The actual intensities of the microbubbles and the noise spots are obtained. The microbubbles spots can then be obtained by removing the pixels, whose echo intensities are lower than the maximum noise spots intensity value $N$, which are then considered to be the noise spots [25]. Therefore, the contrast microbubbles are detected with less influence of noise spots as illustrated in Figure 46.

From the descriptions above, the contrast microbubbles can be detected by using the different characteristics of echo intensity and geometrical dimension between the microbubbles spots and the noise spots. However, the microbubble image quality is still not very satisfactory for the quantitative assessment of MR although it has already been largely improved. The remaining noise spots still largely influence the microbubble patterns.

3.3.2. Linear Relationship Assumptions

For detecting accurately the microbubbles from the inhomogeneous noisy background of the contrast image cavities, further studies on the characteristics of the microbubbles spots and the noise spots have been made. Since the scattered noise and the inhomogeneity of the cardiac chamber background in the echocardiographic images are intrinsically originated from the image formation process and the characteristics of myocardial tissues, thus, the injected contrast microbubbles in the cardiac chambers are
Figure 46. Illustration of an approach of microbubbles detection by using the different characteristics between microbubbles (black) and noise spots (shadowed). (1) noise spots on the cavity baseline of the original contrast-free image; (2) noise smoothed baseline in the cavity of the contrast-free image after a 2x2 lowpass filtering; (3) noise spots and the maximum echo intensity in the cavity of the contrast-free image after subtracting the noise smoothed baseline from the original contrast-free image cavity; (4) microbubbles spots and noise spots on the baseline of the cavity in the unprocessed contrast image; (5) the microbubbles spots and the noise spots in the cavity of the contrast image after subtracting the noise smoothed baseline of the contrast-free image from the contrast image; (6) microbubbles less interfered by noise spots after eliminating the noise spots whose echo intensities are lower than the maximum intensity of the noise spots.
only superposed on the contrast-free images to form the so-called contrast images. We therefore suppose that there exist some relationships between the thresholds for the microbubbles detection in a contrast images cavity and the intensities pertaining to the noise spots and inhomogeneous background in the cavity of the contrast-free image. Let us first define the following characteristics in the contrast-free image.

The average and maximum intensities in an ROI of the contrast-free image cavity is measured as $I_{a1}$ and $I_{m1}$, respectively. This contrast-free image is processed by a $2 \times 2$ low-pass filtering to get a noise smoothed image. By subtracting the noise smoothed image from the original contrast-free image in which any resulting negative pixel value is set to zero, we get the subtracted basal image which only contains the relative value of the noise spots intensities in the ROI of the cavity measured average and maximum intensities as $I_{a2}$ and $I_{m2}$, respectively.

Assuming this relationship between threshold $T$ and combination $I$ of the characteristics $I_{a1}, I_{m1}, I_{a2}$ and $I_{m2}$ is a linear one as denoted in the following formula:

$$ T = c \times I + c_0 \quad (3.1) $$

where $c_0$ and $c$ are linear regressive coefficients between the threshold $T$ and the combination $I$. $I = c_1 \times I_{a1} + c_2 \times I_{m1} + c_3 \times I_{a2} + c_4 \times I_{m2}$. $c_1, c_2, c_3$ and $c_4$ are weight factors of the characteristics contributing to the linear regression.

The characteristic $I_{a1}$ is considered to be able to represent mainly the gross intensity of the inhomogeneous background which may influence generally the simulation of microbubbles; $I_{m1}$ is able to represent the local maximum intensity of the inhomogeneous background which may sensitively simulate the microbubbles appearance; $I_{a2}$ is able to represent the gross intensity of the noise spots in the cavity which is considered the main factor influencing the threshold of microbubbles detection; $I_{m2}$ is able to represent local influence of the main noise spot which may simulate the microbubbles.
Based on the assumptions of linear relationship between the threshold detecting the microbubbles and the four characteristics in the contrast-free image chamber, a good linear regressive relation in each cardiac chamber is obtained next, which has validated the previous assumptions.

3.3.3. Linear Regressive Relationship

In order to build up the relationship between the threshold and the four characteristics described before for the three ROI situations: ES phase for LV, ES phase for LA and ED phase for LA, seventy three cases randomly selected from the 202 image sequences were used as a training set to derive the linear regressive relationship for these situations.

Following the preceding descriptions, the noise smoothed contrast-free image has been subtracted from the five subsequent contrast images, therefore normalizing the contrast images cavities. In order to discriminate accurately the microbubbles from the noise spots, we searched manually for a proper intensity threshold in the histogram of each cavity ROI of the normalized contrast images. The threshold was obtained by observing the elimination of the noise spots from the image cavity in which a compromise was made for the intensity overlaps between the microbubbles and the noise spots, as illustrated in Figure 47. In the same way, five thresholds for each of the three ROI situations of the normalized contrast images were therefore obtained. Since there is usually few changes of the background intensities of the contrast image in the adjacent few heart beats, we used the same threshold \( T \) for an ROI situation of the five normalized contrast images by means of averaging the obtained five thresholds.

By applying the described method for obtaining the average threshold \( T \) and measuring the 4 characteristics \( \text{Im1}, \text{Ia1}, \text{Im2} \) and \( \text{Ia2} \) in the three ROI situations of the 73 independent image sequences training set, we obtained therefore a sequence of data for training the linear regressive relationship. With the help of the REGRESSION procedure in the SPSS statistical package, the linear relationship was analyzed.

In order to build an appropriate linear relationship between the variable \( T \) and variable \( I \) for each ROI among the 73 samples, the weight factor values of the 4
characteristics were adjusted for maximizing the F statistic \[^{[46]}\]. The F statistic serves to test how well the regression model fits the data and it is defined as Equation (3.2).

\[
F = \frac{MS_{regression}}{MS_{residual}}
\]

where the Mean Squares for regression and Mean Squares for residuals (\(MS_{regression}\) and \(MS_{residual}\)) are divided by their degrees of freedom (\(df\)). As illustrated in Equation (3.2), the total observed variability between the sample points \((X_i, Y_i)\) and \((X_j, Y_j)\) is subdivided into two parts: the variability due to regression \((MS_{regression})\) and the variability due to residuals \((MS_{residual})\).

---

**Figure 47.** Illustration of a threshold finding from normalized contrast image. (1) microbubbles (black) and noise spots (shadowed) in the inhomogeneous cavity of the original contrast image; (2) the microbubbles spots and the noise spots in the cavity of the contrast image after being normalized by subtracting the noise smoothed baseline of the contrast-free image; (3) detected microbubbles with few individual and superimposed noise after a compromised thresholding.
\[ F = \frac{\text{Mean Square Regression}}{\text{Mean Square Residual}} \] (3.2)

where the Mean Square Regression and Mean Square Residual are, respectively, \((Y_{pi}-Y_0)^2\) and \((Y_i-Y_{pi})^2\) divided by their degrees of freedom (DF). As illustrated in Figure 48, the total observed variability between the sample point \((X_i, Y_i)\) and the sample means \((X_0, Y_0)\) is subdivided into two components: Regression \((Y_{pi}-Y_0)\) and Residual \((Y_i-Y_{pi})\).

![Diagram with labeled points](Image)

**Figure 48.** Illustration of the components of the F statistic.

When the maximum F statistic was found, the linear regressive relationships were then obtained. The F statistics, standard errors, and linear regressive models for the three ROI situations are described as follows. The overlay plots are illustrated in Figures 49, 50 and 51.
ES phase for LV:

F statistic = 44.49
Standard Error = 8.59

\[
T[LV/ES] = 0.041I[LV/ES]+32.6 \quad (3.3a)
\]

\[
I[LV/ES] = -0.7xIm[LV/ES]+65xIa2[LV/ES]+15xIm2[LV/ES] \quad (3.3b)
\]

Figure 49. Overlay scatter plot of linear regression in the LV ROI of ES image phase
ES phase for LA:

F statistic = 39.64
Standard Error = 9.28

\[ T[\text{LA/ES}] = 0.12xI[\text{LA/ES}] + 27.4 \]  \hspace{1cm} (3.4a)

\[ I[\text{LA/ES}] = -0.2xIa1[\text{LA/ES}] + 0.7xIm1[\text{LA/ES}] \\
+ 60xIa2[\text{LA/ES}] + Im2[\text{LA/ES}] \]  \hspace{1cm} (3.4b)

**Figure 50.** Overlay scatter plot of linear regression in the LA ROI of ES image phase
ED phase for LA:

F statistic = 88.68
Standard Error = 8.19

\[
\begin{align*}
T[\text{LA/ED}] &= 0.13I[\text{LA/ED}]+17 \\
I[\text{LA/ED}] &= -0.5x_Ia_1[\text{LA/ED}]+I_m_1[\text{LA/ED}] \\
&\quad + 5.5x_Ia_2[\text{LA/ED}]+3x_Im_2[\text{LA/ED}] 
\end{align*}
\] (3.5a)

Figure 51. Overlay scatter plot of linear regression in the LA ROI of ED image phase
As described above, the maximum standard error in the three ROI situations is only 9.28 which is small enough for predicting accurately the threshold in the contrast echocardiographic images.

The $F$ statistic values were obtained as 44.5 for LV ROI of ES phase, 39.6 for LA ROI of ES phase, and 88.7 for LA ROI of ED phase. Therefore, the linear regressive model in the LA ROI of ED image phase best fits the data. The reason for this is because the LA chamber in the ED image phase is with less interference other than the scattered noise and inhomogeneous background, such as myocardial tissue presence which may present in the LV and LA chamber of ES image phase. Besides, the quantity of injected microbubbles is usually high which is easier to be analyzed visually. On the other side, it brings about difficulty to find the thresholds between the microbubble pixels and the background noise pixels in the LV and LA chambers of ES image phase where microbubbles often cover the whole background which is not the case in the LA chamber of ED image phase.

In the derived linear combinations of the characteristics, $I_{m1}$ and $I_{a1}$ have the same quantity scale, and the characteristics $I_{m2}$ and $I_{a2}$ have another identical quantity scale. It is then possible to compare influences of the characteristics with the same quantity scale to the derived threshold by analyzing their weight coefficients. From the weight factor values of the characteristics in those combinations, we can see that the maximum echo intensity of an original contrast-free image cavity ROI is more influential than the average one, and the average echo intensity of an subtracted contrast-free image cavity ROI, i.e., noise spots, is more influential than the maximum one. Therefore, the derived linear regressive models have validated the previous assumed significances of the four characteristics.

The adaptive detection models of the contrast microbubbles were therefore built by linear regressive relationships and applied to 202 image sequences.

### 3.3.4. Adaptive Detection of Microbubbles

The microbubbles detection models obtained from the training set of 73 image sequences were used for improving the contrast images quality, therefore, the automated
MR assessment by microbubbles motion analysis is less interfered from scattered noise spots and inhomogeneous intensity background. The image processing results yielded from the microbubbles detection models are described next by an example other than the case from the training set, and the efficiency of the models are validated thereafter by the automated MR assessment results.

As has been described before, the contrast-free image ROIs, as shown in Figure 52, are processed by a 2x2 lowpass filtering as the baseline image. By subtracting the baseline image from the original contrast-free image, the normalized contrast-free image ROIs are obtained which only contain the noise spots shown in Figure 53. The four characteristics are then measured in the ROIs of both images in Figures 52 and 53. Each subsequent contrast image shown in Figure 54 is also normalized by subtracting the baseline image. The microbubbles with noise spots in the ROIs are obtained as shown in Figure 55. The thresholds for detecting the microbubbles from the noise spots in the ROIs of the normalized contrast image are then computed with the relevant linear regressive models. The contrast microbubbles are then detected, as shown in Figure 56, when applying the thresholds to the normalized contrast image ROIs and eliminating the pixels in the ROIs whose echo intensities are below the corresponding thresholds.

Figure 52. An example of an original contrast-free image ROIs
Figure 53. Normalized contrast-free image ROIs consisting of noise spots

Figure 54. An example of the subsequent contrast image ROIs
Figure 55. Normalized contrast image ROIs consisting of the microbubbles and the noise spots

Figure 56. Adaptively detected microbubbles in the contrast image ROIs
From observation of the detected microbubbles image, we conclude that the image processing algorithms we have developed are efficient for improving the image quality for microbubbles visualization, and also for quantitative MR assessment which will be described in detail in the next chapter.
4. MR Pattern Recognition

The goal of our work is to assess quantitatively the mitral regurgitation by analysis and interpretation of the contrast 2-D echocardiograms. After the contrast echocardiogram processing with the approaches described in the previous chapter, the microbubbles in the images appear as ideally restored in noiseless chambers, which are easily recognized by physicians and objectively characterized by means of the computer. However, quantification of mitral regurgitation indicated by the microbubbles remains a major task in MIRAS.

4.1. General Theory of Pattern Recognition

What is pattern recognition? In their widest sense, patterns are the means by which we interpret the physical world. There are two aspects to pattern recognition: developing a decision rule and using it. The actual recognition occurs in the use of rule; the pattern is defined in the learning process by the labeled samples. In mathematical pattern recognition, we want a decision rule which can classify examples of patterns quickly. We may usually proceed with more leisure in learning the pattern, that is, in deriving the decision rule.

The pattern is defined by the labeled samples of that pattern. Samples are presented as examples of one class patterns or another. A pattern recognition problem thus begins with class definitions and labeled samples of those classes in some workable representation. The problem is solved when a decision rule is derived which assigns a unique label to new patterns.
4.1.1. Pattern Recognition Process

The necessary process in deriving the decision rule in a practical pattern recognition problem is indicated diagrammatically in Figure 57. The physical world from which given patterns arise is characterized completely only by its physical embodiment that is essentially infinite in dimensionality. We characterize that embodiment numerically by some set of finite dimensional measurements. We shall refer to the raw data describing the world as the measurement space; that is, a sample of a pattern is represented by specific values of all the measurements, corresponding to a point in the measurement space. The pattern classification algorithms we will study should be applied to a feature space which (a) is finite dimensional, (b) is of relatively low dimension, and (c) contains sufficient information to satisfactorily perform the classification. Feature selection process is thus performed to map the measurement space to a feature space that is of a finite and usually small set of dimensions. Finally, we must develop, on the basis of a limited set of labeled samples, a decision rule with which we can classify a point in the feature space corresponding to an unlabeled sample. The process of deriving the decision rule is called pattern classification. The two major areas of pattern recognition are feature selection and pattern classification procedures.

![Diagram of Pattern Recognition Process]

Figure 57. Stages in the derivation of the decision rule

4.1.2. Basic Methodologies of Pattern Recognition \[48\]

An automatic pattern recognition may be implemented by three principal categories of methodology: heuristic, mathematical, and linguistic or syntactic. It is not uncommon to find a combination of these methods in a pattern recognition system.
The heuristic approach is based on human intuition and experience. The structure and performance of a heuristic system will depend to a large degree on the cleverness and experience of the system designers.

The mathematical approach is based on classification rules which are formulated and derived in a mathematical framework. This is in contrast with the heuristic approach. The mathematical approach may be subdivided into two categories: deterministic and statistical.

The deterministic approach is based on a mathematical framework which does not employ explicitly the statistical properties of the pattern classes under consideration. The deterministic approach is based on iterative algorithms.

The statistical approach is based on mathematical classification rules which are formulated and derived in a statistical framework. The design of a statistical pattern classifier is generally based on the Bayes classification rule and its variations. This rule yields an optimum classifier when the probability density function of each pattern population and the probability of occurrence of each pattern class are known.

Characterization of patterns by primitive elements (subpatterns) and their relationships suggests automatic pattern recognition by the linguistic or syntactic approach. A pattern can be described by a hierarchical structure of subpatterns analogous to the syntactic structure of languages. This permits application of formal language theory to the pattern recognition problem. This approach is particularly useful in dealing with patterns which cannot be conveniently described by numerical measurements or are so complex that local features cannot be identified and global properties must be used.

### 4.1.3. Implementation of Pattern Recognition

Once a specific design method has been selected, we are still faced with the actual design and implementation problem. In most cases, representative patterns from each class under consideration are available. In these situations, supervised pattern recognition techniques are applicable. In a supervised learning environment, the system is "taught" to recognize patterns by means of various adaptive schemes. The essentials of this approach
are a set of training patterns of known classification and the implementation of an appropriate learning procedure.

In some applications, only a set of training patterns of unknown classification may be available. In these situations, unsupervised pattern recognition techniques are applicable. As mentioned above, supervised pattern recognition is characterized by the fact that the correct classification of every training pattern is known. In the unsupervised case, however, we are faced with the problem of actually learning the pattern classes present in the given data.

The learning or training takes place only during the design (or updating) phase of a pattern recognition system. Once acceptable results have been obtained with the training set of patterns, the system is applied to the task of actually performing recognition on samples drawn from the environment in which it is expected to operate. The quality of the recognition performance will be largely determined by how closely the training patterns resemble the actual data with which the system will be confronted during normal operation.

4.1.4. MR as a Pattern Recognition Problem

The quantitative assessment of the mitral regurgitation is a pattern recognition issue. It mainly involves the MR pattern characterization by microbubbles characteristics in each MR class, feature selection for computational cost reduction, classifier selection which minimizes the misclassification rate, and estimate of the selected classifier performance with the finite sample set (202 image sequences in our work).

According to the established criteria, the training set of the 202 image sequences was categorized in the four classes in which 75 cases were from Class 0, 58 cases from Class 1, 28 cases from Class 2, and 41 cases from Class 3. Based on the feature selection from the MR measurement space and minimization of misclassification probability in the training set, an optimal hierarchical classification tree was constructed and its performance was validated in test sets.
4.2. Microbubble Motion Patterns Analysis

Measurement space is originated from the interpretation of the physical world. The measurements of MR are based on the heuristic analysis of visual assessment. Conventional assessment of the presence and severity of MR is performed by evaluating visually the microbubbles flowing patterns namely the variations of microbubbles quantity and distribution in the cardiac chambers, which denote the microbubbles flowing quantity and directions. The MR criteria for visual assessment have also shown that MR degree definition is based on the microbubble flowing patterns, since different regurgitation classes have different flowing quantity and direction.

4.2.1. MR Criteria Interpretation

The criteria for the 4 MR classes can be interpreted in terms of microbubble motion patterns as follows:

![Diagram](image)

**Figure 58.** Illustration of microbubbles motion and distribution in an ES image phase of Class 0. Black spots denote microbubbles. LV = left ventricle; LA = left atrium; PML = posterior mitral leaflet; AML = anterior mitral leaflet.
Class 0 (no regurgitation): There are no microbubbles flowing into LA cavity from LV through mitral valve defects, after a contrast agent is injected in the LV chamber. Therefore the LA cavity is always contrast-free throughout all the cardiac phases, as illustrated in Figure 58.

Class 1 (mild regurgitation): There are a few microbubbles flowing into LA cavity from LV through mitral valve orifices during ST, after a contrast agent is injected in the LV chamber. With the heart beat increase, the microbubbles continue flowing into LA cavity during ST and may flow back to LV during DT or disappear when the agent transit time expires. The microbubbles quantity in LA is much smaller than in LV because there are only some orifices in the mitral valve. Meanwhile, the microbubbles may flow in a concentrated way, which is the so-called jet effect, through the small orifices, as illustrated in Figure 59.

![Diagram](image)

**Figure 59.** Illustration of microbubbles motion and distribution in an ES image phase of Class 1. Black spots denote microbubbles. LV = left ventricle; LA = left atrium; PML = posterior mitral leaflet; AML = anterior mitral leaflet.

Class 2 (moderate regurgitation): There are some microbubbles somewhat homogeneously flowing into LA cavity from LV through mitral valve defects during ST, after a contrast agent is injected in the LV chamber. These microbubbles occupy the whole LA chamber in the first two systoles. With the heart beat increase, the microbubbles continue flowing into the LA cavity during ST and may flow back to LV during DT or disappear when the agent transit time expires. The microbubbles density in
LA, however, is still smaller than in LV because the mitral valve defect is not too severe. Besides, the microbubbles may also flow in a concentrated way by a jet effect through the small orifices in the mitral valve, as illustrated in Figure 60.

![Figure 60](image)

**Figure 60.** Illustration of microbubbles motion and distribution in an ES image phase of Class 2. Black spots denote microbubbles. LV = left ventricle; LA = left atrium; PML = posterior mitral leaflet; AML = anterior mitral leaflet.

![Figure 61](image)

**Figure 61.** Illustration of microbubbles motion and distribution in an ES image phase of Class 3. Black spots denote microbubbles. LV = left ventricle; LA = left atrium; PML = posterior mitral leaflet; AML = anterior mitral leaflet.
Class 3 (severe regurgitation): There are some microbubbles homogeneously flowing into LA cavity from LV through mitral valve defects or unclosed mitral valve during ST, after a contrast agent is injected in the LV chamber. With the heart beat increase, more microbubbles continue flowing into the LA cavity during ST and may flow back to LV during DT or disappear when the agent transit time expires. However, the microbubbles density in the LA cavity is equal to or even gradually larger than that in the LV from the first to nearly the tenth ST, as illustrated in Figure 61.

### 4.2.2. Basic Microbubbles Measures

According to the interpreted criteria above, more microbubbles flow into LA from LV, and their presence in LA lasts more heart beats with the increase of MR degree. The microbubbles quantity is represented by their echo intensity and number of pixels. The jet effect is represented by an inhomogeneous distribution in LA (without the jet effect, there is usually a homogeneous distribution of microbubbles). Therefore, the measurements of the microbubbles motion are based on the measures of the variations of their echo intensity and number of pixels, as well as their distributions in LA influenced by those in LV. For these reasons, the following basic parameters are measured and explained with the illustration in Figure 62.

![Figure 62. Illustration of basic measurements. The dashed regions are LV and LA ROIs; The black spots are microbubbles.](image-url)
Alv - The area of ROI in LV (dashed area in LV cavity in Figure 62). This is the observation area of the LV cavity in which the microbubbles are injected.

Ala - The area of ROI in LA (dashed area in LA cavity in Figure 62). This is the observation area of LA cavity into which the microbubbles flow by mitral regurgitation.

AMLv - The number of microbubbles pixels in LV (total area of the black spots in LV in Figure 62). This is a spatial dimension descriptor of microbubbles quantity in the LV ROI. Usually, a larger number of microbubbles pixels represents a larger quantity of microbubbles when their gross echo intensity is a constant.

AMLa - The number of microbubbles pixels in LA (total area of the black spots in LA in Figure 62). This is a spatial dimension descriptor of microbubbles quantity in the LA ROI. Usually, a larger number of microbubbles pixels represents a larger quantity of microbubbles when their gross echo intensities are constants. It may also measure the jet effect when the microbubbles cover fewer pixels but with higher echo intensities demonstrating the microbubbles concentration.

IMLv - The sum of echo intensities of microbubbles in LV (total echo intensities of the black spots in LV in Figure 62). This is an echo intensity descriptor of microbubbles quantity in the LV ROI. Higher microbubbles echo intensities usually correspond to a higher quantity of microbubbles when their number of pixels is a constant. It is an important descriptor of the microbubbles quantity in the LV ROI, because the microbubbles usually occupy the whole LV chamber, therefore, the microbubbles spatial dimension descriptor AMLv is influenced by the LV ROI area Alv.

IMLa - The sum of echo intensities of microbubbles in LA (total echo intensities of the black spots in LA in Figure 62). This is an echo intensity descriptor of microbubbles quantity in the LA ROI. Higher microbubbles echo intensities usually correspond to a higher quantity of microbubbles when their number of pixels is a constant. It may also measure the jet effect when the microbubbles have higher echo intensities but cover fewer pixels demonstrating the microbubbles concentration.

SDLa - The standard deviation of microbubbles echo intensity spatial distribution in LA. Homogeneous distribution of microbubbles corresponds to a larger standard
deviation, and inhomogeneous distribution corresponds to a smaller standard deviation. Therefore it may describe the existence of a jet effect in the LA ROI.

For calculating SDla, the 256x256 pixel image region was divided into 6x6 pixel cells. The total intensity of each cell was used as a point involved in the calculation. SDla is assigned zero when IMla is calculated to be zero, i.e., there are no microbubbles in LA. This case happens in Class 0 or other classes when a false-negative is present. Since SDla represents microbubbles spatial distribution in LA, SDla value is theoretically largest in the LA of Class 3 case images. In Class 2 case images, there are some scattered microbubbles and also a concentration of microbubbles caused by jet effect, thus SDla in this class is smaller. SDla in Class 1 is the smallest (a positive value) in the positive MR cases since there is a jet effect with less scattered microbubbles in the LA cavity. From the tendency of SDla variation with the degree of MR, we therefore define SDla as zero.

4.2.3. Microbubble Motion Measurements

Based on the 7 basic measures of the contrast echocardiographic images, sixty eight descriptors of microbubble motion patterns for each image sequence, denoted by M(n), consisting of ES and ED phases of the 5 heart beats were studied and represented in the following equations, where i denotes ES and ED, and j denotes the heart beat:

\[ M(i)[i][j] = IMla[i][j] \] (4.1)

denotes the total microbubbles echo intensities in the ROI of LA, as illustrated with the spots in LA ROIs in Figure 63. It describes the time evolution of the echo intensities in the LA when microbubbles quantity changes.

Contrast image phases

![Contrast image phases](image)

Figure 63. Illustration of microbubbles intensities in LA ROIs of ES contrast image phases.
M(2)[i][j] = AMla[i][j]  \hspace{2cm} (4.2)

denotes the microbubbles covered area in the ROI of LA, as illustrated with the black spots in LA ROIs in Figure 64. It describes the time evolution of the covered area in the LA when the microbubbles quantity changes.

**Figure 64.** Illustration of microbubbles covered area in LA ROIs of ES contrast image phases.

M(3)[i][j] = IMla[i][j] / Ala[i][j]  \hspace{2cm} (4.3)

denotes the distribution density of microbubbles echo intensities in the ROI of LA. As illustrated in Figure 65, the total intensities in the LA ROI are divided by the dashed area. It describes the time evolution of the intensity density in the LA ROI when the microbubbles quantity changes.

**Figure 65.** Densities of microbubbles intensities in LA ROIs of ES contrast image phases.
M(4)[i][j] = AMLa[i][j] / Ala[i][j]  \hfill (4.4)

denotes the distribution density of microbubbles covered area in the ROI of LA. As illustrated in Figure 66, the total area of the black spots in the LA ROI is divided by the dashed area. It describes the time evolution of the covered area density in the LA ROI when the microbubbles quantity changes.

Contrast image phases

heart beat 1  heart beat 2  ....  heart beat 5

Figure 66. Densities of microbubbles covered area in LA ROIs of ES contrast image phases.

M(5)[i][j] = IMLa[i][j] / AMLa[i][j]  \hfill (4.5)

denotes the microbubbles density in their covered area in the ROI of LA. As illustrated in Figure 67, the total intensities of the spots in the LA ROI are divided by the total circle area. It may reflect the concentration of the microbubbles in the LA ROI, measuring a jet effect.

Contrast image phases

heart beat 1  heart beat 2  ....  heart beat 5

Figure 67. Densities of microbubbles intensities in their covered area in LA ROIs of ES contrast image phases.
denotes the total microbubbles echo intensities in the ROI of LA normalized by that in the ROI of LV in ES. As illustrated in Figure 68, the total intensities of the spots in the LA ROI are divided by those in the LV ROIs. It describes the time evolution of the echo intensities in the LA ROI compared with that in the LV ROI. This descriptor is relatively independent from the absolute echo values and therefore insensitive to physical variations of the image formation process.

\[
M(6)[i][j] = M(1)[i][j] / IMlv[ES][j] = IMla[i][j] / IMlv[ES][j] \quad (4.6)
\]

\[
M(7)[i][j] = M(2)[i][j] / AMlv[ES][j] = AMla[i][j] / AMlv[ES][j] \quad (4.7)
\]

denotes the number of microbubbles pixels in the ROI of LA normalized by that in the ROI of LV in ES. As illustrated in Figure 69, the total area of the black spots in the LA ROI is divided by that in the LV ROIs. It describes the time evolution of the microbubbles covered area in the LA ROI compared with that in the LV ROI. This descriptor is relatively insensitive to the microbubbles quantity injected in the LV.

\[
\text{Figure 68. Illustration of microbubbles intensities in LA ROIs normalized by those in LV ROIs of ES contrast image phases.}
\]

\[
\text{Figure 69. Illustration of microbubbles covered area in LA ROIs normalized by that in LV ROIs of ES contrast image phases.}
\]
MIRAS for Contrast 2-D Echocardiograms

\[ M(8)[i][j] = M(3)[i][j] / (IMlv[ES][j]/Alv[ES][j]) \]
\[ = (IMla[i][j]/Ala[i][j]) / (IMlv[ES][j]/Alv[ES][j]) \]  

(4.8)

denotes the distribution density of microbubbles echo intensities in the ROI of LA normalized by that in the ROI of LV in ES, as illustrated in Figure 70. It describes the time evolution of the intensities density in the LA ROI compared with that in the LV ROI. This descriptor is relatively independent from the absolute echo values and therefore insensitive to physical variations of the image formation process.

**Contrast image phases**

![Image of heart beats 1 to 5 with microbubbles changes](image)

*Figure 70. Densities of microbubbles intensities in LA ROIs normalized by those in LV ROIs of ES contrast image phases.*

\[ M(9)[i][j] = M(4)[i][j] / (AMlv[ES][j]/Alv[ES][j]) \]
\[ = (AMla[i][j]/Ala[i][j]) / (AMlv[ES][j]/Alv[ES][j]) \]  

(4.9)

denotes the distribution density of microbubbles covered area in the ROI of LA normalized by that in the ROI of LA in ES, as illustrated in Figure 71. It describes the time evolution of the covered area density in the LA ROI compared with that in the LV ROI. This descriptor is relatively insensitive to the microbubbles quantity injected in the LV.

**Contrast image phases**

![Image of heart beats 1 to 5 with microbubbles changes](image)

*Figure 71. Densities of microbubbles covered area in LA ROIs normalized by those in LV ROIs of ES contrast image phases.*
MIRAS for Contrast 2-D Echocardiograms MR Pattern Recognition

\[
M(10)[i][j] = M(5)[i][j] / (IMlv[ES][j]/AMLv[ES][j]) \\
= (IMla[i][j]/AMLla[i][j]) / (IMlv[ES][j]/AMLv[ES][j]) 
\tag{4.10}
\]

denotes the microbubbles density in their covered area in the ROI of LA normalized by that in the ROI of LV in ES, as illustrated in Figure 72. It may reflect the concentration of the microbubbles in the LA ROI compared with that in the LV ROI, measuring a jet effect.

**Contrast image phases**

![Diagram](image)

*Figure 72. Microbubbles densities in their covered area in LA ROIs normalized by those in LV ROIs of ES contrast image phases.*

\[
M(11)[i][j] = SDla[i][j] 
\tag{4.11}
\]

denotes the standard deviation of microbubbles spatial distribution in LA ROI. As illustrated in Figure 73, the standard deviation is measured different for different concentration of black spots in the LA ROIs. It describes the inhomogeneity of microbubbles spatial distribution caused by the jet effect in the LA ROI.

**Contrast image phases**

![Diagram](image)

*Figure 73. Microbubbles spatial concentration in LA ROIs of ES contrast image phases.*
M(k+6)[j] = 1/2 (∑ₐ M(k)[i][j]) ; k=6, 7, ..., 11  \hspace{1cm} (4.12)

describes the average value of M(k) in a cardiac cycle, for compensating the false-negative caused by jet effect.

M(k+17)[i][j] = M(k)[i][j] - M(k)[i][jj] ; k=1, 2, ..., 11  \hspace{1cm} (4.13)

describes the change of the microbubble amount and distribution between adjacent cardiac cycles j and jj (jj=j+4, if j<=1; jj=j-1, if j>1) of M(k).

M(k+17)[j] = M(k)[j] - M(k)[jj] ; k=12, 13, ..., 17  \hspace{1cm} (4.14)

describes the change of the microbubble amount and distribution between adjacent cardiac cycles j and jj (jj=j+4, if j<=1; jj=j-1, if j>1) of M(k).

M(k+34)[i][j] = 1/2 (M(k)[i][j] + M(k)[i][jj]) ; k=1, 2, ..., 11  \hspace{1cm} (4.15)

describes the average value in two adjacent cardiac cycles j and jj, for compensating the randomness of microbubble motion, (jj=j+4, if j<=1; jj=j-1, if j>1) of M(k).

M(k+34)[j] = 1/2 (M(k)[j] + M(k)[jj]) ; k=12, 13, ..., 17  \hspace{1cm} (4.16)

describes the average value in two adjacent cardiac cycles j and jj, for compensating the randomness of microbubble motion, (jj=j+4, if j<=1; jj=j-1, if j>1) of M(k).

M(k+51)[i][j] = 1/3 (M(k)[i][j] + M(k)[i][jj] + M(k)[i][jjj]) ; k=1, 2, ..., 11  \hspace{1cm} (4.17)

describes the average value in three adjacent cardiac cycles j, jj and jjj, for compensating the randomness of microbubble motion, (jj=j+4, if j<=1; jj=j-1, if j>1; jjj=j+3, if j<=2; jjj=j-2, if j>2) of M(k).

M(k+51)[j] = 1/3 (M(k)[j] + M(k)[jj] + M(k)[jjj]) ; k=12, 13, ..., 17  \hspace{1cm} (4.18)
describes the average value in three adjacent cardiac cycles $j, j_j$ and $j_{jj}$, for compensating the randomness of microbubble motion, $(j= j+4, \text{ if } j \leq 1; \ j_j=j-1, \text{ if } j > 1; \ j_{jj}=j+3, \text{ if } j \leq 2; \ j_{jj}=j-2, \text{ if } j > 2)$ of $M(k)$.

In the calculation of Equation (4.5), $M(5)$ is assigned zero when $AM\text{La}$ is calculated to be zero, i.e., there are no microbubble in LA. In Equations (4.6) through (4.10), the descriptors are assigned to zero when $AM\text{Lv}$ is zero ($IM\text{Lv}$ is simultaneously zero), i.e., no microbubbles are observed in the LV ROI.

The MR measurement space is therefore obtained consisting of the 18 mathematical equations (4.1) - (4.18). However, the number of MR measurements is too large to be entirely used for classifier design in the personal computer system and for the limited number of image sequences as a training set. MR feature selection is then an important phase for the implementation of the automatic pattern recognition system design.

4.3. Theory of Classification Rule

Feature selection and classification largely depend on a certain classification rule. In the following, we introduce two basic classification rules, which will be used in the MR classification problem, the Bayes linear classification rule and the hierarchical linear classification rule.

4.3.1. Bayes Linear Classification Rule [47] [75]

Bayes classification rule aims at minimizing average loss or risk for each class. We define the loss function $C(w_k/w_i)$, where $k, i = 1, 2, ..., K$, is the loss incurred when a pattern actually belonging to $w_i$ is placed in class $w_k$. The conditional average loss is defined as

$$L(x, w_k) = \sum_{i=1}^{K} C(w_k/w_i) \ p(w_i/x), \quad (4.19)$$
where \( P(w_i | x) \) is the conditional probability of a pattern vector \( x \) belonging to class \( w_i \). Thus this function is conditioned upon the pattern vector \( x \) and can be interpreted as the loss associated with measuring vector \( x \) and assigning it to various classes \( w_k \) each weighted by a loss incurred by that particular classification. If \( L(x, w_k) \) is minimized, then the classifier becomes statistically optimum in a Bayes sense and is therefore often referred to as a Bayes classifier. To minimize the conditional average loss the classifier must assign \( x \) to the category \( w_k \) where \( L(x, w_k) \leq L(x, w_i) \) for all \( i = 1, 2, ..., K \). Thus \( L(x, w_i) \) must be calculated for all classes and we will therefore end up with \( K \) conditional average losses. An obvious set of discriminant functions then becomes

\[
g_k(x) = -L(x, w_k). \tag{4.20}
\]

A possible solution is obtained by using Bayes rule:

\[
P(w_i | x) = \frac{P(w_i) P(x | w_i)}{P(x)}, \tag{4.21}
\]

where \( P(w_i) \) is the a priori probability of class \( w_i \) whose crude estimate can be obtained from the training set samples:

\[
P(w_i) = \frac{M_i}{\sum_{i=1}^{K} M_i}. \tag{4.22}
\]

Therefore

\[
L(x, w_k) = \sum_{i=1}^{K} C(w_k | w_i) P(w_i) P(x | w_i) / P(x). \tag{4.23}
\]

But \( P(x) \) is common to all conditional average losses (for each class) and consequently plays no role in the discriminant function. Therefore let

\[
I(x, w_k) = \sum_{i=1}^{K} C(w_k | w_i) P(w_i) P(x | w_i). \tag{4.24}
\]

We will simply refer to \( I(x, w_k) \) as a form of conditional average loss noting that the \( P(x) \) statistic is missing. In terms of discriminant functions
\[ g_k(x) = -l(x, w_k), \] \hspace{1cm} (4.25)\\

and the decision rule states that we place \( x \) in class \( w_k \) such that \( g_k(x) \geq g_i(x), \ (l(x, w_k) \leq l(x, w_i)) \), for all \( i = 1, 2, ..., K \). This decision rule has become known as the Bayes rule and the classifier which implements such a decision surface is known as a Bayes classifier.

A type of loss function which might be of interest could be described by

\[
C(w_k/w_i) = \begin{cases} 
-\lambda_i & i = k \\
0 & i \neq k 
\end{cases} \hspace{1cm} (4.26)
\]

where \(-\lambda_i > 0\). Such a loss function can be interpreted as assigning a negative loss (or positive gain) to a correct decision and no loss (gain) to an incorrect decision. Thus it is possible to weigh the importance of guessing one class correctly over another class by assigning different values to \( \lambda_i \). The modified conditional average loss then becomes

\[
l(x, w_k) = - \sum_{i=1}^{K} \lambda_i \delta(i - k) p(w_i)p(x/w_i) \hspace{1cm} (4.27)
\]

where \( \delta(i - k) \) is the Kronecker delta function. Thus

\[
l(x, w_k) = - \lambda_k p(w_k)p(x/w_k) \hspace{1cm} (4.28)
\]
or

\[
g_k(x) = \lambda_k p(w_k)p(x/w_k). \hspace{1cm} (4.29)
\]

This implies a decision rule such that \( x \) is placed in class \( w_k \) when

\[
\lambda_k p(w_k)p(x/w_k) \geq \lambda_i p(w_i)p(x/w_i), \hspace{1cm} i = 1, 2, ..., K. \hspace{1cm} (4.30)
\]

We assume the distribution in the feature space according to the Gaussian or normal distribution:

\[
p(x) = (1/((2\pi)^{N/2}|\Phi|^{1/2})) \exp[-(1/2)(x - \mu)^\top \Phi^{-1}(x - \mu)] \hspace{1cm} (4.31)
\]
where $\Phi$ is a positive definite matrix with elements $\sigma_{ij}$. $\Phi^{-1}$ is the inverse of $\Phi$. $|\Phi|$ is the determinant of $\Phi$. We use $\mathcal{X}'$ to indicate the transpose of the row vector $\mathcal{X}$. $\mu$ and $\Phi$ are mean of the distribution and covariance matrix, respectively.

The mean and covariance of a density $p(\mathcal{X})$ are defined as

$$
\mu = \mathcal{E}(\mathcal{X}) = \int_{\mathcal{X}} \mathcal{X} \ p(\mathcal{X}) \ d\mathcal{X}
$$

(4.32a)

and

$$
\sigma_{ij} = \mathcal{E}((x_i - \mu_i)(x_j - \mu_j)) = \int_{\mathcal{X}} (x_i - \mu_i)(x_j - \mu_j) \ p(\mathcal{X}) \ d\mathcal{X},
$$

(4.32b)

where $\mathcal{X} = (x_1, ..., x_N)$, $\mu = (\mu_1, \mu_2, ..., \mu_N)$, and $\mathcal{E}[\cdot]$ is the expected value operator.

We may estimate the covariance and the mean of the distribution from $M$ sample points by the equations

$$
\mu = \frac{1}{M} \sum_{j=1}^{M} x_j,
$$

(4.33a)

$$
\sigma_{ij} = \frac{1}{M} \sum_{k=1}^{M} (x_{ki} - \mu_i)(x_{kj} - \mu_j).
$$

(4.33b)

We can also calculate the mean and covariance in an iterative manner, updating an estimate of these parameters [47]. If $\mu_i(M)$ and $\sigma_{ij}(M)$ are the mean and covariance estimated by the first $M$ samples, we note that

$$
\sigma_{ij}(M) = \frac{1}{M} \sum_{k=1}^{M} x_{ki} x_{kj} - \mu_i(M)\mu_j(M)
$$

(4.34)

by expanding the product in (4.33b) and defining

$$
S_{ij}(M) = \frac{1}{M} \sum_{k=1}^{M} x_{ki} x_{kj}.
$$

We now note that
\[ \mu_i(M+1) = \frac{1}{(M+1)} (\mu_i(M) + x_{M+1,i}) \]  
\[ (4.35a) \]

\[ S_{ij}(M+1) = \frac{1}{(M+1)} (S_{ij}(M) + x_{M+1,i} x_{M+1,j}) \]  
\[ (4.35b) \]

and

\[ \sigma_{ij}(M+1) = S_{ij}(M+1) - \mu_i(M+1) \mu_j(M+1) ; \]  
\[ (4.35c) \]

Equations (4.35) provide us with an iterative algorithm for calculation of the parameters, given successive samples. Once having calculated the parameters, we have the distribution completely specified.

In this classification problem, we assume that the samples in each class \( w_k \) \((k = 1, ..., K)\) have a normal distribution. Thus, the conditional probability of a pattern vector \( x \) belonging to class \( w_k \) is given by:

\[ p(x/w_k) = \left(\frac{1}{(2\pi)^{N/2} |\Phi_k|^{1/2}}\right) \exp\left(-\frac{1}{2} (x - \mu_k)^T \Phi_k^{-1} (x - \mu_k)\right). \]  
\[ (4.36) \]

By using Equation (4.29), the discriminant function can be transformed to be

\[ g_k(x) = \log \left\{ \gamma_{ik} p(w_k) p(x/w_k) \right\} . \]  
\[ (4.37) \]

By substituting Equation (4.36) into Equation (4.37), we obtain

\[ g_k(x) = \log \left\{ \gamma_{ik} p(w_k) \right\} - (N/2) \log 2\pi - (1/2) \log |\Phi_k| \]
\[ - (1/2)((x - \mu_k)^T \Phi_k^{-1} (x - \mu_k)) . \]  
\[ (4.38) \]

However the term \((N/2) \log 2\pi\) is common to all such discriminant functions and consequently can be removed. Thus

\[ g_k(x) = -(1/2) x \Phi_k^{-1} x' + x \Phi_k^{-1} \mu_k' - (1/2) \mu_k \Phi_k^{-1} \mu_k' \]
\[ + \log \left\{ \gamma_{ik} p(w_k) \right\} - (1/2) \log |\Phi_k| . \]  
\[ (4.39) \]

Supposing that the covariance matrix \( \Phi_k \) of each class is the same, which is usually a reasonable assumption, the linear discriminant function becomes
\[ g_k(x) = x \Phi^{-1} \mu_k' - (1/2) \mu_k \Phi^{-1} \mu_k' + \log \{ \gamma_k \mathbf{P}(w_k) \} \] (4.40a)

or

\[ g_k(x) = x \Phi^{-1} \mu_k' - (1/2) \mu_k \Phi^{-1} \mu_k' + \log \mathbf{P}'(w_k) \] (4.40b)

where \( \mathbf{P}'(w_k) = \gamma_k \mathbf{P}(w_k) \). When the loss function value \( \gamma_k \) is assumed as 1, which is an usual assumption, we thus obtain the linear discriminant function

\[ g_k(x) = x \Phi^{-1} \mu_k' - (1/2) \mu_k \Phi^{-1} \mu_k' + \log \mathbf{P}(w_k). \] (4.41)

Here the common terms, \(-(1/2) x \Phi_k^{-1} x'\), and \(-(1/2) \log |\Phi|\) have been eliminated and the discriminant function becomes linear. Therefore, the Bayes linear classification rule is obtained as Equations (4.40) and (4.41). The posteriori probability of class \( w_k \) in the linear classifier is therefore estimated by

\[ \mathbf{P}(w_k|x) = \exp[ g_k(x) ] / \sum_{k=1}^{K} \exp[ g_k(x) ]. \] (4.42)

### 4.3.2. Hierarchical Linear Classification Rule

In the usual approach to classification, a common set of features is used jointly in a single decision step. A different approach is to use a multistage or sequential hierarchical decision scheme. The rationale for hierarchical classifiers is that by using different subsets of features at the various decision levels better performance may be achieved than by employing a single "best" set of features in a one-step decision \[^{[49]}\]. Classification trees offer an effective implementation of such hierarchical classifiers. Indeed, classification trees have become increasingly important due to their conceptual simplicity and their computational efficiency. A large variety of methods have been proposed for the design of classification trees and have been successfully applied to solve a wide range of pattern recognition problems \[^{[50] - [60]}\].

In the design of a hierarchical classification tree, rejection of a class at intermediate stages as not likely candidates is only suboptimal in the Bayes minimum cost sense \[^{[52]}\], which is only used in some reason such as inability to handle large data, limitations on the number of classes a particular stage can handle, etc. The following optimal decision
rule for a hierarchical classifier can be obtained from a consequence of Bayes' theorem [58].

A hierarchical classification tree \( T \) is composed of a root node \( L_1 \), a set of intermediate nodes \( \{ L_i \} \), and terminal nodes labeled with class labels \( \{ w_k \} \) (Figure 74). The sample to be classified undergoes a sequence of tests on the path from the root to a terminal node, at which point it is classified as belonging to that class.

![Figure 74. A hierarchical classification tree T](image)

The probability of correct recognition of a random sample using the classification tree will be derived under the following three constraints on the model:

1) The terminal node labels which denote classes are unique, i.e., no class label occurs at more than one terminal.

2) If \( f_i \) denotes the feature set used at node \( L_i \), the decision rule at the node is only a function of that feature set.

3) If \( f_1, f_2, f_3, ..., f_n \) are the feature sets used on the path to the class label \( w_k \) at a terminal, these feature sets are statistically independent conditioned on that class, i.e.,

\[
p(x_1, x_2, x_3, ..., x_n/w_k) = \prod_{i=1}^{n} p(x_i/w_k) \quad (4.43)
\]

where \( x_i \) is the observed value vector of feature set \( f_i \).
Then if \( P_c(w_k/L_i) \) represents the probability of making a correct decision at node \( L_i \) for a random sample pattern from class \( w_k \), it follows from the above assumptions that the correct classification probability for that class is

\[
P_c(w_k) = \prod_{L_i \in S(w_k)} P_c(w_k/L_i) \tag{4.44}
\]

where \( S(w_k) \) denotes the set of nodes on the path to class \( w_k \) in \( T \).

Hence, considering the loss function and the \textit{a priori} probability, the correct recognition rate of the tree is the weighted sum of products, viz.,

\[
P_c(T) = \sum_k P'(w_k) \cdot \prod_{L_i \in S(w_k)} P_c(w_k/L_i) \tag{4.45}
\]

One can also define the correct recognition rate of a node \( P_c(L_i) \), averaged across all the classes reachable from it, denoted by \( \Omega(L_i) \) as

\[
P_c(L_i) = \sum_{w_k \in \Omega(L_i)} P'(w_k) \cdot \frac{P_c(w_k/L_i)}{\sum_{w_k \in \Omega(L_i)} P'(w_k)} \tag{4.46}
\]

As a consequence of Bayes' theorem, it follows that \( P_c(L_i) \) is maximized by using a maximum-likelihood rule. Thus, if \( L_{i1}, L_{i2}, ..., L_{in} \) are the immediate descendant nodes of \( L_i \) and \( x_i \) the observed feature vector of feature set \( f_i \), the optimal decision would be to traverse node \( L_i^* \), where

\[
\sum_{w_k \in \Omega(L_i^*)} P'(w_k) \cdot P(x_i/w_k) = \max_{1 \leq j \leq n} \sum_{w_k \in \Omega(L_{ij})} P'(w_k) \cdot P(x_i/w_k) \tag{4.47}
\]

By applying the Bayes' linear classification rule (4.40) or (4.41) to each node, we obtain therefore the hierarchical linear classification rule.
4.4. MR Feature Selection

Probably the most important aspect of pattern recognition is that of feature selection. There are many feature selection methods which may be applied to particular problems. However, low dimensionality and sufficient information of the selected features are generally required.

4.4.1. Previous Studies on Feature Selection

Pattern recognition research has considered various questions concerning the relationship between the limited size of the training set, the number of features, and the estimation of some performance criterion. A number of authors, including El-Sheikh and Wacker [61], have investigated the optimal number of features for a given finite design sample size in order to combat the "peaking phenomenon," the apparent loss of classifiability which accompanies an increase in the number of features without an increase in the number of training samples. An excellent review of work done in this area is presented in Jain and Chandrasekaran [62]. Another group of authors has looked into the effect of the ratio of training sample size to feature set dimensionality on the expected performance of an empirically designed classifier used on the true test distribution. Raudys and Pikelis [63] catalog the development of a number of approximate expressions for the expected performance of the linear classifier and an exact expression for the quadratic classifier. Asymptotic expansions for the quadratic classifier have also been developed by Han [64] and McLabchlan [65]. Unfortunately, these expressions are too complex to provide valuable insight and their accuracy has not been experimentally verified. Thus, the relationship between sample size and dimensionality has been inferred through simulation [63], the investigation of related criteria [66] [67], and a look at the performance of these classifiers tested on the training set [68]. However, Fukunaga and Hayes [69] provided valuable insight into the relationship between the number of features and the number of training samples, and their results have been experimentally verified.
4.4.2. Sample Size and Feature Size

Previous studies have shown that the classification error depends on the structure of a classification algorithm, asymptotic probability of misclassification, and the ratio of learning sample size per class $M$ to feature size $N$ for all linear discriminant functions [63]. In order to derive a classification algorithm and evaluate substantially the classifier performance, the ratio $M/N$ was studied and found that the training set error is an extremely biased estimate of either the Bayes or test set error rate if the ratio of sample size per class to feature size $M/N$ is less than three [68]. Besides, Foley [68] also demonstrated that the estimate bias between training and test sets is very small when $M/N$ is larger than 4.

In our particular problem, the term "sample" is one of the 202 image sequences that were labeled in the 4 classes. There were 75 samples labeled Class 0, 58 samples labeled Class 1, 28 samples labeled Class 2, and 41 samples labeled Class 3. All the 202 samples were used as a training set. The features selected to derive the classification structure should also satisfy the relation between sample size and feature size for a less biased estimate.

4.4.3. Feature Selection by $F$ Statistic

Features selected should be able to distinguish the differences between classes. In order to measure the similarity between the probability density functions of two classes $i$ and $j$, which are supposed as normal distributions with equal covariance matrix $\Phi$, the Mahalanobis distance is defined as

$$D_{ij} = (\mu_i - \mu_j)^T \Phi^{-1} (\mu_i - \mu_j)$$

(4.48)

where $\mu_i$ and $\mu_j$ are the mean vectors of Class $i$ and $j$. The smaller the distance, the greater the similarity.

$F$ statistic, which should be noted different from that in Equation (3.2) of Section 3.3.3, is a test of two class means which is based on Mahalanobis distance and is defined as
MIRAS for Contrast 2-D Echocardiograms  MR Pattern Recognition

\[ F = D_{ij} (n - 1 - p) \frac{n_i n_j}{(p (n - 2)(n_i + n_j))} \]  (4.49)

where \( D_{ij} \) is the Mahalanobis distance, \( p \) is the number of variables from which features are selected, \( n \) is the total number of samples, \( n_i \) and \( n_j \) are, respectively, the sample size in Class \( i \) and \( j \).

When the \( F \) statistic is the criterion for feature selection, the \( F \) values between pairs of classes are calculated first. The variable that has the largest \( F \) for the two classes that are closest is selected as a potential feature.

SPSS advanced statistics package\(^{[34]}\) which provides a stepwise variable selection algorithm, including forward entry, stepwise selection, and backward elimination, the \( F \) statistic can be used as a criterion. In the stepwise method the first variable included in the analysis is entered, the value of the \( F \) statistic is reevaluated for all variables not in the model, and the variable with the largest acceptable \( F \) value is entered next. At this point, the variable entered first is reevaluated to determine whether it meets the removal \( F \) value. If it does, it is removed from the model.

4.4.4. Classification Tree Selection

Since the database we used as the training set is from the assessment results by contrast 2D echocardiography in Table 2, we assume that the number of patients in each MR class we used represent the patient population. Thus the \textit{a priori} probabilities of the four classes are obtained in Equations (4.50).

\[ P(0) = 0.45; \quad P(1) = 0.22; \quad P(2) = 0.16; \quad P(3) = 0.17 \]  (4.50)

By using the SPSS advanced statistics package and the \( F \) statistic as the feature selection criterion and considering the \textit{a priori} probabilities in Equations (4.50), the overall correct classification rate of each promising classifier was calculated bases on the Bayes linear classification rule given by Equation (4.41). In order to adapt the SPSS package calculation, a suboptimal hierarchical classification rule was used for feature selection and classifier construction. In calculation of classification rates for feature selection in the following trees, each node was considered as an isolated one, i.e., rejecting a descendant node of a lower probability at a intermediate node. The
classification rate at a terminal node was obtained by multiplying each ascendant node classification rate till the root node. Thus we get some potentially useful features and classification trees.

Features for the Bayes linear classifier were selected. Taking into consideration the feature size and sample size, 6 features were selected which resulted in poor classification rates, namely, 51% for Class 0, 52% for Class 1, 39% for Class 2, and 61% for Class 3. Therefore, the average correct classification rate was only 51%, as shown in Figure 75.

![Diagram](image)

Figure 75. Bayes linear classifier and results

Since the number of samples in Class 2 is only 28 which limits the feature size in the single node linear classification, namely, the Bayes linear classification. Therefore, hierarchical linear classification trees were considered to be useful so that more features could be used for several nodes.

The a priori probability values of the 4 classes have shown that P(0) is the largest one of all. We, therefore, considered that Class 0 would be the class to be distinguished first. Besides, Class 1 should also be classified at a more ascendant node for its high a priori probability. However, Class 3 as Class 0 is an extreme class which might also be distinguished first for its clinical significance. Meanwhile, Class 2 should be the last one to be classified because of its smallest sample size. Therefore, the following principal classification trees (Figures 76, 77, and 78) were built and studied as the most promising ones, and features for each node were selected with the help of SPSS package.
Figure 76. One structure of a hierarchical linear classifier and results

Figure 77. One structure of a hierarchical linear classifier and results
Comparing the total correct classification rates of the three principal classifier candidates, we can judge that the hierarchical linear classification tree in Figure 78 is the best one to be used. Besides, other possible classification tree structures were also tested and the total correct classification rates were found to be inferior to the existing ones. Therefore, the hierarchical classification tree in Figure 78 was selected as the classifier for the 4-class MR classification.

### 4.4.5. Features for the Classifier

In the established hierarchical linear classification tree, 6 features were selected for the first node (M/N > 12 in Class 0; M/N > 21 in pooled Class 1, 2 and 3), 7 features for the second node (M/N > 8 in Class 1; M/N > 9 in pooled Class 2 and 3), and 6 features for the third node (M/N > 4 in Class 2; M/N > 6 in Class 3). These features at each node are listed as follows in the order of descendant contribution:
Features at the first node:  

<table>
<thead>
<tr>
<th>Feature</th>
<th>Expression</th>
<th>Standard Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>$F_1 = M(38)[ES][3] = \frac{1}{2} (M(4)[ES][3] + M(4)[ES][2])$</td>
<td>2.32095</td>
</tr>
<tr>
<td>F2</td>
<td>$F_2 = M(37)[ES][3] = \frac{1}{2} (M(3)[ES][3] + M(3)[ES][2])$</td>
<td>-2.20993</td>
</tr>
<tr>
<td>F3</td>
<td>$F_3 = M(39)[ES][5] = \frac{1}{2} (M(5)[ES][5] + M(5)[ES][4])$</td>
<td>0.53201</td>
</tr>
<tr>
<td>F4</td>
<td>$F_4 = M(39)[ED][5] = \frac{1}{2} (M(5)[ED][5] + M(5)[ED][4])$</td>
<td>0.37037</td>
</tr>
<tr>
<td>F5</td>
<td>$F_5 = M(49)[3] = \frac{1}{2} (M(15)[3] + M(15)[2])$</td>
<td>0.37007</td>
</tr>
<tr>
<td>F6</td>
<td>$F_6 = M(41)[ES][1] = \frac{1}{2} (M(7)[ES][1] + M(7)[ES][5])$</td>
<td>-0.31994</td>
</tr>
</tbody>
</table>

Features at the second node:  

<table>
<thead>
<tr>
<th>Feature</th>
<th>Expression</th>
<th>Standard Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>$F_1 = M(49)[1] = \frac{1}{2} (M(15)[1] + M(15)[5])$</td>
<td>2.45098</td>
</tr>
<tr>
<td>F2</td>
<td>$F_2 = M(48)[1] = \frac{1}{2} (M(14)[1] + M(14)[5])$</td>
<td>-2.15448</td>
</tr>
<tr>
<td>F3</td>
<td>$F_3 = M(4)[ED][5]$</td>
<td>1.08513</td>
</tr>
<tr>
<td>F4</td>
<td>$F_4 = M(3)[ED][5]$</td>
<td>-0.77794</td>
</tr>
<tr>
<td>F5</td>
<td>$F_5 = M(44)[ES][4] = \frac{1}{2} (M(10)[ES][4] + M(10)[ES][3])$</td>
<td>0.59202</td>
</tr>
<tr>
<td>F6</td>
<td>$F_6 = M(48)[4] = \frac{1}{2} (M(14)[4] + M(14)[3])$</td>
<td>0.36758</td>
</tr>
</tbody>
</table>

Features at the third node:  

<table>
<thead>
<tr>
<th>Feature</th>
<th>Expression</th>
<th>Standard Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>$F_1 = M(16)[2]$</td>
<td>0.61968</td>
</tr>
<tr>
<td>F2</td>
<td>$F_2 = M(49)[4] = \frac{1}{2} (M(15)[4] + M(15)[3])$</td>
<td>0.59564</td>
</tr>
<tr>
<td>F5</td>
<td>$F_5 = M(45)[ES][3] = \frac{1}{2} (M(11)[ES][3] + M(11)[ES][2])$</td>
<td>0.28889</td>
</tr>
</tbody>
</table>

The contribution of each feature is demonstrated by the standard discriminant function coefficient. The standard discriminant function coefficients, derived by the SPSS advanced statistics package, are used when the features are standardized to a mean of 0 and a standard deviation of 1. Since the features are correlated, it is not possible to assess the importance of an individual variable. The value of the coefficient for a particular
feature depends on the other features included in the function. However, the magnitudes of the coefficients can indicate the relative importance of features. Features with larger coefficients are thought to contribute more to the overall discriminant function.

4.4.5.1. Interpretation of Features at Node 1

The relative contributions of the 6 features at the first node demonstrate that Feature 1 and Feature 2 are the most important features for distinguishing Class 0 from other classes since they have very high absolute values of the standard discriminant function coefficients (2.32095 and 2.20993). Feature 1, namely, \( M(38)[ES][3] \) is the average value between \( M(4)[ES][3] \) and \( M(4)[ES][2] \). Feature 2 is \( M(37)[ES][3] \) which is the average value between \( M(3)[ES][3] \) and \( M(3)[ES][2] \). These two features give the following information:

The distribution densities of microbubbles covered area and microbubbles total echo intensity in LA are most efficient for the recognition of the presence or not of the microbubbles in LA, since the parameters \( M(4) \) and \( M(3) \), as in Equations (4.4) and (4.3), are the direct representations of microbubbles quantity. Besides, the microbubbles in LA mainly present at ES phase other than at ED in the positive MR cases, therefore, distinguished from the always contrast-free image in Class 0. In addition, the most easily distinguished MR presence is in the second and the third cardiac cycles after contrast injection.

Feature 3 and Feature 4 at the first node, namely, \( M(39)[ES][5] \) and \( M(39)[ED][5] \) consist of the average values between \( M(5)[5] \) and \( M(5)[4] \) at, respectively, ES and ED. The parameter \( M(5) \), in Equations (4.5), represents microbubbles echo intensity in their covered area, thus, measuring jet effect. Since the jet effect never happens in Class 0, these two features are also used for the recognition of Class 0 from the others. However, they are much less important than the first two features because their standard coefficients are, respectively, 0.53201 and 0.37037.

Feature 5 at the first node is denoted by

\[
M(49)[3] = \frac{1}{2} (M(15)[3] + M(15)[2]) \\
= \frac{1}{2} [\frac{1}{2} (M(9)[ES][3] + M(9)[ED][3]) + \frac{1}{2} (M(9)[ES][2] + M(9)[ED][2])],
\]
where $M(9)[i][j] = M(4)[i][j] / (AMlv[ES][j]/Alv[ES][j])$. Since it also represents the distribution density of microbubbles covered area having some correlation with Feature 1, it is less important.

Feature 6 at the first node represents $1/2 (M(7)[ES][1] + M(7)[ES][5])$, as in Equation (4.7), measuring the average value of the number of microbubble pixels in LA normalized by that in LV in ES phase of the first and the fifth heart beat images. It is the least important at the first node.

4.4.5.2. Interpretation of Features at Node 2

It is more complicated to distinguish Class 1 from Class 2 and 3 because there is a mild MR in Class 1 and the quantity of microbubbles in LA varies with the heart beats. MR in Class 1 may be confused with that in Class 2 in some cardiac cycles, especially, since both have jet effect.

Feature 1 and Feature 2 at the second node are the most important features for distinguishing Class 1 from Class 2 and 3. Feature 1 is actually the average between

$$1/2 (M(9)[ES][1] + M(9)[ED][1]) \text{ and } 1/2 (M(9)[ES][5] + M(9)[ED][5]),$$

where $M(9)[i][j] = M(4)[i][j] / (AMlv[ES][j]/Alv[ES][j])$. Feature 2 is the average between

$$1/2 (M(8)[ES][1] + M(8)[ED][1]) \text{ and } 1/2 (M(8)[ES][5] + M(8)[ED][5]),$$

where $M(8)[i][j] = M(3)[i][j] / (AMlv[ES][j]/Alv[ES][j])$.

As described in Equations (4.9) and (4.8), it is demonstrated that the distribution densities of microbubbles covered area and microbubbles total echo intensity in LA normalized by those in LV are the most important factors for distinguishing Class 1 from Class 2 and 3 since the microbubbles quantity in Class 1 is the main characteristic being different from those in Class 2 and 3. Besides, both features need the average values between ES and ED which shows the necessity of compensating for the false-negative MR presence at ES caused by the jet effect. The average values of the characteristics
between the first and the fifth heart beats may combine the influence of jet effect usually observed in the beginning heart beat with the normal presence of microbubble quantity in the last observed heart beat, therefore, representing the general characteristic of Class 1.

Feature 3 and Feature 4 at the second node, i.e., \( M(4)[\text{ED}][5] \) and \( M(3)[\text{ED}][5] \) represent the distribution densities of microbubbles covered area and microbubbles total echo intensity in LA which are measured at ED of the fifth heart beat, therefore, reducing the influence of the same jet effect characteristic. The importance of these two features is inferior to that of the first two.

Feature 5 at the second node represents the average value between \( M(10)[\text{ES}][4] \) and \( M(10)[\text{ED}][3] \), where \( M(10)[i][j] = M(5)[i][j] / (IMlv[\text{ES}][j]/AMlv[\text{ES}][j]) \). This feature measures the microbubbles echo intensity density in their covered area in LA normalized by that in LV, which is influenced by the jet effect distinguishing the different characteristic between Class 1 and Class 3.

The equation of Feature 6 at the second node can be expanded as

\[
M(48)[4] = \frac{1}{2} \left( M(14)[4] + M(14)[3] \right) \\
= \frac{1}{2} \left[ \frac{1}{2} \left( M(8)[\text{ES}][4] + M(8)[\text{ED}][4] \right) + \frac{1}{2} \left( M(8)[\text{ES}][3] + M(8)[\text{ED}][3] \right) \right],
\]

where \( M(8)[i][j] = M(3)[i][j] / (IMlv[\text{ES}][j]/AMlv[\text{ES}][j]) \), as described in Equation (4.8), denotes the microbubbles echo intensity density in LA normalized by that in LV. Therefore, Feature 6 measures the average value between ES and ED, and also the average between third and fourth heart beats.

Feature 7 at the second node is the least important feature for classifying Class 1 from Class 2 and 3. It can be denoted by \( SDla[\text{ES}][5] - SDla[\text{ES}][4] \), measuring the variation of microbubbles spatial distribution, namely, the jet effect at ES between the fifth and fourth heart beats.

4.4.5.3. Interpretation of Features at Node 3

Contributions of the features at node 3 are somewhat homogeneous (the standard coefficients varying from 0.22800 to 0.61968) in comparison with those at the first two
nodes. It is easier to distinguish Class 2 from Class 3 since there are some apparently different characteristics between them. There is a presence of jet effect in Class 2 which is never present in Class 3. Meanwhile, the microbubbles quantity in LA in Class 3 is much larger than in Class 2.

The most important features at the third node is the average value between $M(10)[ES][2]$ and $M(10)[ED][2]$, where $M(10)[i][j] = M(5)[i][j] / (IMlv[ES][j]/AMLv[ES][j])$, therefore measuring the jet effect during the second heart beat, as described in Equation (4.10).

Feature 2 at the third node represents

$$M(49)[4] = 1/2 (M(15)[4] + M(15)[3])$$
$$= 1/2 [1/2 (M(9)[ES][4] + M(9)[ED][4]) + 1/2 (M(9)[ES][3] + M(9)[ED][3])],$$

where $M(9)[i][j] = M(4)[i][j] / (AMLv[ES][j]/AMLv[ES][j])$ denotes the density of microbubbles covered area in LA normalized by that in LV. Therefore, Feature 1 measures the average value of microbubble quantity flowed into LA from LV between ES and ED, and also the average between third and fourth heart beats. Therefore, this feature distinguishes the different microbubble quantity flowed to LA between Class 2 and Class 3.

Feature 3 at the third node represents the change of microbubbles spatial distribution in LA at ES $(SDla[ES][3] - SDla[ES][2])$ between the second and the third heart beats, therefore, measuring the jet effect variation. The absolute value of the variation is large in Class 2 but small in Class 3.

Feature 4 can be denoted by

$$1/2 (M(5)[ES][4] + M(5)[ED][4]) / (IMlv[ES][4]/AMLv[ES][4]) - 1/2 (M(5)[ES][3] + M(5)[ED][3]) / (IMlv[ES][3]/AMLv[ES][3]),$$

and Feature 6 can be denoted by

$$M(5)[ES][5] / (IMlv[ES][5]/AMLv[ES][5]) - M(5)[ES][4] / (IMlv[ES][4]/AMLv[ES][4]).$$
Therefore, as described in Equation (4.5), the basic factor behind these two features are the microbubbles echo intensity density in their covered area in LA normalized by that in LV, measuring the jet effect. Feature 4 represents therefore the change between the third and fourth heart beats in the average factor of ES and ED. Feature 6 represents the change of the factor between the fifth and fourth heart beats.

Feature 5 denoted by 1/2 (SD[LA][ES][3] + SD[LA][ES][2]) also measures the jet effect by the average value of the standard deviation of microbubbles spatial distribution in LA at ES between the second and the third heart beats.

In conclusion, the interpretation of the features used at each node have demonstrated the clinical significance and validated the MR criteria of visual assessment. The efficiency of the features are also validated by the automated classification results described in the next chapter.

4.5. Classification Results and Validation

The hierarchical linear classification tree was built by minimizing its total error rate based on the training set of the 202 image sequences (samples), from which the features were selected and interpreted in the preceding chapter. With the limited number of samples, the classification results are demonstrated satisfactory to the clinical reality and the true performance of the classifier is estimated.

4.5.1. Classification Results in Training Set

As a clinical tool, the overall correct classification rate of the classifier is not the only estimate of its performance. Other clinical parameters, such as the individual class classification rate, the sensitivity and specificity, should also be taken into account. With the a priori probabilities in Equations (4.50) and the Bayes linear classification rule in Equation (4.41), i.e., let $\gamma_k = 1, k = 1, 2, ..., K$, we could obtain correct classification rate of 65.1% and partially correct classification rate of 87.5%, and also sensitivities of
61.4% and of 85.8%, specificities of 85.3% and of 94.7% respectively for correct classification and partially correct classification. However the correct classification rates for Class 1 and Class 2 were only 43.1% and 32.1%, respectively. Therefore, we could not obtain a satisfactory performance without using proper loss function values.

For this reason, experiments of classification were performed for choosing satisfactory \( \eta_k \). We obtained the following loss function values

\[
\eta_0 = 0.667; \quad \eta_1 = 1.136; \quad \eta_2 = 1.75; \quad \eta_3 = 1.0
\]

By using the definition \( P'(\omega_k) = \eta_k P(\omega_k) \), we obtain the following values for Equation (4.40b).

\[
P'(0) = 0.30; \quad P'(1) = 0.25; \quad P'(2) = 0.28; \quad P'(3) = 0.17 \quad (4.51)
\]

Therefore, the classification results can be calculated according to the optimal hierarchical linear classification rule, as described in Section 4.3.2. For this selected classifier in Figure 78, the probabilities of feature vectors \( x_1 \), \( x_2 \) and \( x_3 \), respectively at node 1, 2 and 3, in all classes \( (C_0, C_1, C_2, C_3) \) and intermediate nodes, i.e., the pooled classes \( (C_{1,2,3} \) and \( C_{2,3} \) were calculated and compared as follows.

The \( P'(\omega_k) \) values of the classes and nodes (pooled classes) can be calculated as

\[
P'(C_0) = P'(0) \quad (4.52a)
\]

and

\[
P'(C_{1,2,3}) = P'(1) + P'(2) + P'(3) \quad (4.52b)
\]

for node 1;

\[
P'(C_1) = \frac{P'(1)}{P'(1) + P'(2) + P'(3)} \quad (4.52c)
\]

and

\[
P'(C_{2,3}) = \frac{P'(2) + P'(3)}{P'(1) + P'(2) + P'(3)} \quad (4.52d)
\]

for node 2;

\[
P'(C_2) = \frac{P'(2)}{P'(2) + P'(3)} \quad (4.52e)
\]

and

\[
P'(C_3) = \frac{P'(3)}{P'(2) + P'(3)} \quad (4.52f)
\]
for node 3,
we can calculate the linear Discriminant function value \( g_k \) of feature vectors \( x_1, x_2 \) and \( x_3 \) in Equation (4.40b). By substituting \( g_k \) into the following equations, we then obtain the correct classification probabilities for the classes and the intermediate nodes:

\[
P(C_0/x_1) = \exp[g_0(x_1)] / (\exp[g_0(x_1)] + \exp[g_{1,2,3}(x_1)]); \tag{4.53a}
\]

\[
P(C_{1,2,3}/x_1) = \exp[g_{1,2,3}(x_1)] / (\exp[g_0(x_1)] + \exp[g_{1,2,3}(x_1)]); \tag{4.53b}
\]

\[
P(C_1/x_2) = \exp[g_1(x_2)] / (\exp[g_1(x_2)] + \exp[g_{2,3}(x_2)]); \tag{4.53c}
\]

\[
P(C_{2,3}/x_2) = \exp[g_{2,3}(x_2)] / (\exp[g_1(x_2)] + \exp[g_{2,3}(x_2)]); \tag{4.53d}
\]

\[
P(C_2/x_3) = \exp[g_2(x_3)] / (\exp[g_2(x_3)] + \exp[g_3(x_3)]); \tag{4.53e}
\]

\[
P(C_3/x_3) = \exp[g_3(x_3)] / (\exp[g_2(x_3)] + \exp[g_3(x_3)]); \tag{4.53f}
\]

Considering Equation (4.44), we obtain the correct classification probability for each class:

\[
P(C_0/x_1) = P(C_0/x_1); \tag{4.54a}
\]

\[
P(C_1/x_1, x_2) = P(C_{1,2,3}/x_1) \cdot P(C_1/x_2); \tag{4.54b}
\]

\[
P(C_2/x_1, x_2, x_3) = P(C_{1,2,3}/x_1) \cdot P(C_{2,3}/x_2) \cdot P(C_2/x_3); \tag{4.54c}
\]

\[
P(C_3/x_1, x_2, x_3) = P(C_{1,2,3}/x_1) \cdot P(C_{2,3}/x_2) \cdot P(C_3/x_3); \tag{4.54d}
\]

The optimal classification was decided by the class that had largest probability in Equations (4.54).

With the features selected for the hierarchical linear classifier in Figure 78 and the \( P'(w_k) \) values, the training set of the 202 image sequences was classified with Equations (4.40), (4.52), (4.53) and (4.54). The following results were obtained as shown in Table 3.
Table 3. Classification results in the training set of 202 image sequences

Considering the ambiguity of both visual assessment, as indicated in Table 2, and clinical importance between adjacent classes, the dashed regions can be clinically assigned to the same class without significant loss. From Table 3 we can easily obtain that the correct classification rates and partially correct classification rates (dashed regions) are as follows:

<table>
<thead>
<tr>
<th></th>
<th>Correct rates</th>
<th>Part. corr. rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0:</td>
<td>66.7%</td>
<td>89.3%</td>
</tr>
<tr>
<td>Class 1:</td>
<td>67.2%</td>
<td>93.1%</td>
</tr>
<tr>
<td>Class 2:</td>
<td>57.1%</td>
<td>89.3%</td>
</tr>
<tr>
<td>Class 3:</td>
<td>68.3%</td>
<td>97.6%</td>
</tr>
<tr>
<td>Overall Rates:</td>
<td>64.4%</td>
<td>91.7%</td>
</tr>
<tr>
<td>SEN:</td>
<td>89.0%</td>
<td>96.9%</td>
</tr>
<tr>
<td>SPE:</td>
<td>66.7%</td>
<td>89.3%</td>
</tr>
</tbody>
</table>

The correct classification rates have shown that they are meaningful for all classes. The partially correct classification results have shown that they are much higher than the
correct ones and quite satisfactory to clinical needs. The sensitivity and specificity for correct classification and partially correct classification are also clinically significant.

However, the mentioned classification results were obtained in the training set of 202 image sequences with which the classifier was designed. It is dangerous to evaluate the classifier performance only in a training set since it may give an optimistically biased estimate of the true performance of the classifier\(^{68}\). Therefore, the classifier should also be evaluated in test sets so as to achieve a realistic estimate of the classifier performance and to provide clinical confidence.

### 4.5.2. Estimation Methods of Classifier Performance

Estimation of the expected performance of a classifier is an important, yet difficult problem in pattern recognition. A number of testing procedures have been proposed and are widely used. In the holdout method, a number of the original samples are withheld from the design process. This provides an independent test set, but drastically reduces the size of the training set. In the resubstitution method, the classifier is tested on the original training samples, as we did in Section 4.5.1. This maintains the size of the training set, but ignores the independent issue generating a dangerously optimistic performance estimate\(^{68}\). The leave-one-out method\(^{70}\) is designed to alleviate these difficulties. It avoids drastically dividing the available sample set into training and test as in the holdout method, while maintaining an independence between them. Thus, the procedure utilizes all available samples more efficiently, and produces a conservative error estimate. However, the leave-one-out method has much greater variance than the resubstitution method. Besides, it requires excessive computation. The rotation method\(^{35}\) can combat these disadvantages of the leave-one-out method.

More recently, Efron\(^{71}\) proposed a resampling procedure, called the bootstrap method, in which artificial samples are generated from the existing samples, and the optimistic bias between the resubstitution error and the classifier error when tested on independent samples is estimated from them.

The analysis of these techniques has been a popular pattern recognition research topic. Novak\(^{72}\) presents a method of computing the error of a specific classifier, given the parameters of the test distribution. Raudys and Pikelis\(^{63}\) give an excellent review of
work done in approximating the expected performance in the parametric case and provide explicit expressions for several empirically designed classifiers. Toussaint\[^{35}\] catalogs these and other testing methods and gives an overview of some of the early associated work. More recent work is surveyed by Hand\[^{73}\]. Fukunaga and Hayes\[^{74}\] apply an error expression to the various methods of error estimation, and offer a unified and comprehensive approach to the analysis of classifier performance.

### 4.5.3. Rotation Method

The classifier performance can be estimated by using the resubstitution method and the rotation method simultaneously to obtain upper and lower bounds of the true performance, since the upper bound obtained in the training set is an optimistically biased estimate, and the lower bound obtained in the test set is a conservatively biased estimate. The upper bounds of the true performance of the established classifier have been obtained to be 64.4\% for correct classification rate and 91.7\% for partially correct classification rate in Section 4.5.1. Here, we will discuss the lower bounds of estimate with the rotation method.

The rotation method\[^{35}\] is a compromise that is less biased than the holdout method and less computationally loaded than the leave-one-out method. Therefore, it is a method well suited to medium-sized data sets as in our problem.

Let \( \{X,\theta\} = \{X_1, \theta_1; X_2, \theta_2; \ldots; X_N, \theta_N\} \) be the set of \( N \) samples available, where \( X_i \) and \( \theta_i \) denote, respectively, the measurement information and the label or classification information of the \( i \)th sample. It is assumed that each \( \theta_i \) associated with \( X_i \) is the correct label, i.e., the samples have been correctly preclassified. Then the rotation method procedure can be described as the following steps.

1) Take a small subset of \( P \) samples as a test set from the total \( N \) samples

\[
\{X,\theta\}_i^{TS} = \{X_1, \theta_1; X_2, \theta_2; \ldots; X_P, \theta_P\}
\]

such that \( 1 \leq P < N \) and \( N/P \) is an integer, \( P/N \leq 1/2 \). Then the training set is

\[
\{X,\theta\}_i^{TR} = \{X_{P+1}, \theta_{P+1}; X_{P+2}, \theta_{P+2}; \ldots; X_N, \theta_N\}.
\]
2) Train the classifier on \( \{X, \theta\}_i^{TR} \).

3) Test the classifier on \( \{X, \theta\}_i^{TS} \) to obtain a proportion of errors denoted by \( P_e[\Pi]_i \).

4) Do steps 1) - 3) for \( i = 1, 2, ..., N/P \) such that \( \{X, \theta\}_i^{TS} \) and \( \{X, \theta\}_j^{TS} \) are disjoint for \( i = 1, 2, ..., N/P, j=1, 2, ..., N/P, \) and \( i \neq j \).

5) The resulting estimate of \( P_e \) is computed as

\[
P_e = \frac{P}{N} \sum_{i=1}^{N/P} P_e[\Pi]_i.
\] 

\[ (4.55) \]

4.5.4. **Estimation Results in Test Sets**

According to the procedure mentioned above, the data set of the 202 samples was divided into three test sets each of which had 67, 67, and 68 samples. Thus, \( N/P \approx 3 \). By combining each two set of the samples used for training the classifier, the classifier performance was estimated by the rest test sets as the matrixes in Tables 4, 5 and 6. Statistical results, including individual classification rate for each class, overall classification rate, sensitivity, and specificity, of each classified test data were listed in the following of each matrix.
### Predicted Classes

<table>
<thead>
<tr>
<th>Actual Classes</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>19</td>
<td>8</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 4. Result from test set 1 by using the rotation method

#### Statistics of Table 4:

<table>
<thead>
<tr>
<th></th>
<th>Correct rates</th>
<th>Part. corr. rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0:</td>
<td>67.9%</td>
<td>96.4%</td>
</tr>
<tr>
<td>Class 1:</td>
<td>83.3%</td>
<td>91.7%</td>
</tr>
<tr>
<td>Class 2:</td>
<td>30.8%</td>
<td>84.6%</td>
</tr>
<tr>
<td>Class 3:</td>
<td>50.0%</td>
<td>85.7%</td>
</tr>
<tr>
<td>$P_e{\Pi}_1$:</td>
<td>58.3%</td>
<td>90.1%</td>
</tr>
<tr>
<td>SEN:</td>
<td>87.2%</td>
<td>89.7%</td>
</tr>
<tr>
<td>SPE:</td>
<td>67.9%</td>
<td>96.4%</td>
</tr>
</tbody>
</table>
## Predicted Classes

<table>
<thead>
<tr>
<th>Actual Classes</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>19</td>
<td>7</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>10</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 5. Result from test set 2 by using the rotation method

### Statistics of Table 5:

<table>
<thead>
<tr>
<th>Class</th>
<th>Correct rates</th>
<th>Part. corr. rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0:</td>
<td>59.4%</td>
<td>81.3%</td>
</tr>
<tr>
<td>Class 1:</td>
<td>62.5%</td>
<td>100.0%</td>
</tr>
<tr>
<td>Class 2:</td>
<td>50.0%</td>
<td>87.5%</td>
</tr>
<tr>
<td>Class 3:</td>
<td>63.6%</td>
<td>100.0%</td>
</tr>
<tr>
<td>$P_e[\Pi]_2$:</td>
<td>58.3%</td>
<td>90.9%</td>
</tr>
</tbody>
</table>

SEN: 85.7% 97.1%
SPE: 59.4% 81.3%
### Predicted Classes

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actual Classes</strong></td>
<td><strong>#class</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>11</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>17</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 6. Result from test set 3 by using the rotation method

### Statistics of Table 6:

<table>
<thead>
<tr>
<th>Class</th>
<th>Correct rates</th>
<th>Part. corr. rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0</td>
<td>73.3%</td>
<td>93.3%</td>
</tr>
<tr>
<td>Class 1</td>
<td>56.7%</td>
<td>96.7%</td>
</tr>
<tr>
<td>Class 2</td>
<td>42.9%</td>
<td>71.4%</td>
</tr>
<tr>
<td>Class 3</td>
<td>75.0%</td>
<td>87.5%</td>
</tr>
<tr>
<td>$P_e[Π]_3$:</td>
<td>60.9%</td>
<td>87.0%</td>
</tr>
</tbody>
</table>

SEN: 75.5% 92.5%
SPE: 73.3% 93.3%
By averaging the above results of the test sets and substituting the above $P_{\epsilon}[\Xi]_i$ values ($i = 1, 2, 3$) to Equation (4.55), we obtain the lower bounds of the classification estimates and the average values of sensitivity and specificity of the test sets as follows.

**Average Test Results:**

<table>
<thead>
<tr>
<th>Class</th>
<th>Correct rates</th>
<th>Part. corr. rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0:</td>
<td>66.8%</td>
<td>90.3%</td>
</tr>
<tr>
<td>Class 1:</td>
<td>67.5%</td>
<td>96.0%</td>
</tr>
<tr>
<td>Class 2:</td>
<td>41.2%</td>
<td>81.2%</td>
</tr>
<tr>
<td>Class 3:</td>
<td>63.0%</td>
<td>91.5%</td>
</tr>
</tbody>
</table>

$P_{\epsilon}:$  
59.2%  
89.3%

SEN:  
82.8%  
93.1%

SPE:  
66.9%  
90.3%

The overall classification estimates in the training set (64.4% and 91.7%) and in the test set (59.2% and 89.3%) show that their differences are mild because the proper consideration of sample size and feature size was taken in the training of the hierarchical linear classifier. Thus both training and test estimates are less biased. By averaging the upper and lower bounds of the classification estimates, and averaging the sensitivity and specificity values in the training set and mean test sets, we obtain the true performance of the developed classifier as follows:
### Classifier True Performance Estimates:

<table>
<thead>
<tr>
<th>Class</th>
<th>Correct rates</th>
<th>Partial. corr. rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0:</td>
<td>66.8%</td>
<td>89.8%</td>
</tr>
<tr>
<td>Class 1:</td>
<td>67.4%</td>
<td>94.6%</td>
</tr>
<tr>
<td>Class 2:</td>
<td>49.2%</td>
<td>85.3%</td>
</tr>
<tr>
<td>Class 3:</td>
<td>65.7%</td>
<td>94.3%</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Rates:</td>
<td>61.8%</td>
<td>90.5%</td>
</tr>
<tr>
<td>SEN:</td>
<td>85.9%</td>
<td>95.0%</td>
</tr>
<tr>
<td>SPE:</td>
<td>66.8%</td>
<td>89.8%</td>
</tr>
</tbody>
</table>

The comparison of the overall classification rates, sensitivities and specificities between the visual assessment results obtained from Table 2 in the Introduction chapter and the automated assessment results are listed in the following:

<table>
<thead>
<tr>
<th>Visual Assessment</th>
<th>Automated Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>corr.</td>
<td>part. corr.</td>
</tr>
<tr>
<td>Overall Rates:</td>
<td>76.3%</td>
</tr>
<tr>
<td>Sensitivity:</td>
<td>97.0%</td>
</tr>
<tr>
<td>Specificity:</td>
<td>69.0%</td>
</tr>
</tbody>
</table>

The data show that the automated assessment results are not too much inferior to the visual ones, therefore, the developed classifier performance is quite satisfactory.

Finally, MIRAS can automatically give the probabilities of a case being classified in the 4 classes in terms of the Equations (4.54), therefore, helping users to analyze the assignment of a case to a class in an quantitative way.
4.5.5. Examples of Classification

Let us take some examples to show how a case is classified. The products between the \textit{a priori} probabilities and the loss function values were given in Equation (4.51) as follows:

\[ P'(0)=0.30; \quad P'(1)=0.25; \quad P'(2)=0.28; \quad P'(3)=0.17 \]

According to Equations (4.52), we have \( P'(\omega_k) \) values at each node:

\[ P'(C_0) = 0.3 \quad \text{and} \quad P'(C_{1,2,3}) = 0.7 \quad \text{for node 1}, \]
\[ P'(C_1) = 0.36 \quad \text{and} \quad P'(C_{2,3}) = 0.64 \quad \text{for node 2}, \]
\[ P'(C_2) = 0.62 \quad \text{and} \quad P'(C_3) = 0.38 \quad \text{for node 3}. \]

4.5.5.1. Example 1

For one of the cases, the following conditional probabilities at the three nodes were computed in terms of Equations (4.53).

\[ P(C_0/X_1) = 0.8022 \quad \text{and} \quad P(C_{1,2,3}/X_1) = 0.1978 \quad \text{at node 1}, \]
\[ P(C_1/X_2) = 0.4090 \quad \text{and} \quad P(C_{2,3}/X_2) = 0.5910 \quad \text{at node 2}, \]
\[ P(C_2/X_3) = 0.7367 \quad \text{and} \quad P(C_3/X_3) = 0.3633 \quad \text{at node 3}. \]

By using Equations (4.54) the following class probabilities are then computed and displayed by menus:

\[ \begin{align*}
 P(C_0/X_1) &= 0.8022 \\
 P(C_1/X_1, X_2) &= 0.0809 \\
 P(C_2/X_1, X_2, X_3) &= 0.0861 \\
 P(C_3/X_1, X_2, X_3) &= 0.0308
\end{align*} \]
Therefore MIRAS automatically classifies this case to Class 0. The user is also confident to accept the automated classification result since the probability of Class 0 is the largest and is much larger than other ones.

4.5.5.2. Example 2

For another case, the following conditional probabilities at the three nodes were computed:

\[
P(C_0/x_1) = 0.4565 \quad \text{and} \quad P(C_{1,2,3}/x_1) = 0.5435 \quad \text{at node 1},
\]

\[
P(C_1/x_2) = 0.9152 \quad \text{and} \quad P(C_{2,3}/x_2) = 0.0848 \quad \text{at node 2},
\]

\[
P(C_2/x_3) = 0.9651 \quad \text{and} \quad P(C_3/x_3) = 0.0349 \quad \text{at node 3}.
\]

By using Equations (4.54) the following class probabilities are then computed and displayed by menus:

\[
P(C_0/x_1) = 0.4565
\]
\[
P(C_1/x_1, x_2) = 0.4972
\]
\[
P(C_2/x_1, x_2, x_3) = 0.0445
\]
\[
P(C_3/x_1, x_2, x_3) = 0.0016
\]

MIRAS classifies this case to Class 1. The user can observe that although the final classification is Class 1, there exists an important probability of the case belonging to Class 0, i.e., this case is quite near Class 0. Thus this is an ambiguous case between Class 0 and Class 1.

4.5.5.3. Example 3

For one more case, the following conditional probabilities at the three nodes were computed:

\[
P(C_0/x_1) = 0.3131 \quad \text{and} \quad P(C_{1,2,3}/x_1) = 0.6869 \quad \text{at node 1},
\]
MIRAS for Contrast 2-D Echocardiograms  

\[ P(C_1/x_2) = 0.1678 \quad \text{and} \quad P(C_{2,3}/x_2) = 0.8322 \quad \text{at node 2}, \]

\[ P(C_2/x_3) = 0.1093 \quad \text{and} \quad P(C_3/x_3) = 0.8907 \quad \text{at node 3}. \]

By using Equations (4.54) the following class probabilities are then computed and displayed by menus:

\[ P(C_0/x_1) = 0.3131 \]
\[ P(C_1/x_1, x_2) = 0.1153 \]
\[ P(C_2/x_1, x_2, x_3) = 0.0623 \]
\[ P(C_3/x_1, x_2, x_3) = 0.5092 \]

MIRAS classifies this case to Class 3. The user can observe that the second possible classification is Class 0. However, Class 0 and Class 3 are two extreme classes which can not be possibly assigned to the same case. Therefore this case may be misclassified and needs user's visual assessment. However, this misclassification situation does not usually happen.
5. Conclusion

Mitral regurgitation assessment by contrast 2-D echocardiography was found to be a harmless and efficient technique, and was validated intraoperatively by Dr. Guerreiro in a work conducted at the main Oporto hospital [13] [19]. The MIRAS system presented in this work has implemented a quantitative method for automated MR assessment in contrast 2-D echocardiograms, overcoming the usually encountered problems in visual assessment, namely, high level of background noises interfering and confusing microbubbles presence, time consuming observation, demand of high skilled clinical experience from the observer, subjective and qualitative results varying with different observers, etc.

For designing the MIRAS system, 202 image sequences were used which were captured from contrast echocardiograms of multi-injected 157 patients. Multi-injections of contrast agent were performed since it was difficult to do visual assessment in some of these images which are poor not only because of noise but also of false MR appearance in some cardiac cycles, i.e., false-negative or false-positive MR presence. The use of all these poor image sequences meant that we had to face difficult assessment problems and the implemented MIRAS was therefore shown to be flexible enough to be used in a wide range of image conditions.

In the 4-class MR assessment, MIRAS, based on a low cost personal computer, has shown a quite satisfactory performance of 61.8% correct classification rate and 90.5% partially correct classification rate even with the poor quality contrast 2-D echocardiograms. MIRAS has, therefore, the main qualities to represent an useful clinical aid in MR assessment.

MIRAS works in an automatic way with a reasonably low degree of human intervention. Thus it can be operated by a non-expert without any computer training or experience to evaluate the mitral regurgitation in an image case within about 20 minutes.
working on a 12 MHz PC/AT computer. It is also easy to use the MIRAS software according to the hints in the software menu of each operation step. Besides, MIRAS can help users analyze class ambiguity in a particular case by showing the closeness between the probabilities of optimal decision class and suboptimal decision class, thus, giving an objective evidence of MR classification. Values of features used for classification can also be shown in order to help the users to understand the mitral regurgitation criteria in a quantitative way.

With the increase of patient image cases being assessed, the database of MIRAS, including the prevalences (a priori probabilities) and the 4-class MR characteristics, are able to be updated by an authorized doctor with the new data for promoting the system’s reliability. For this purpose MIRAS uses the formula in Equations (4.35) to update the mean and covariance presented in Section 4.3.

Although MIRAS is a promising clinical tool aiding in the quantitative MR assessment, there are still some shortcomings to be overcome in further research. The following potential solutions are proposed.

In the image acquisition process, the algorithm for spatial alignment between images of different cardiac cycles by R-wave triggering can be further refined since the developed real-time R-wave detection procedure sometimes may cause false detection when only strongly noisy ECG image is available. An efficient algorithm for R-wave detection from the blurred ECG image is able to speed up the image acquisition process and also to decrease the manual control.

Misrecognition between microbubble pixels and myocardial tissue pixels greatly influences the microbubbles motion analysis, therefore, decreasing the performance of MR assessment. What we propose is the further study of different characteristics able to discriminate these two different pixel sets after some image transformations. Besides, an efficient algorithm for accurate detection of endocardial borders other than manual tracing of LA and LV cavities would improve the automation of ROIs definition process.

Microbubble detection from noisy cavities is one of the most important operations influencing MR assessment. Although the existing algorithms can well remove major noise, thus, improving microbubble image quality, the remaining noise may still simulate some microbubbles presence. Further study on possible transformations of contrast-free
and contrast images in order to find out more characteristics influencing the microbubbles detection thresholds could be of use. Quadratic relationships between them could also be taken into consideration, because of the nonlinearity of the image formation system.

The number of image frames used for the assessment is an important factor influencing the overall speed of the automated MR assessment. Further experience is still needed in order to find out and eventually confirm our choice of five heart beats with two frames per heart beat.

Besides of its utility in MR assessment, MIRAS is able to be adapted to solve some problems in other contrast echocardiograms, such as aortic valve regurgitation assessment, tricuspid valve regurgitation assessment, and the presently popular interesting issue of myocardial perfusion image analysis.

The image acquisition and alignment procedure in MIRAS is generic to all these applications as long as the system initial options are chosen. The automatic spatial alignment algorithm can perform effectively in different applications by defining a proper matching region.

ROIs definition methods can be used to define ROIs in right ventricle, right atrium and aorta cavities. The developed hand-tracing tools are suitable to all ROIs definition issues, especially useful in myocardial perfusion images in which contrast regions are more difficult to be automatically distinguished from the myocardial tissues.

The adaptive detection of contrast microbubbles in this work is based on the analysis of the relation between the microbubble pixels intensity and the noise pixels intensity in a contrast-free image cavity. This analysis method and the derived linear relationships can be applied to the contrast microbubbles detection in any kind of contrast images as long as the parameters of the linear relationships are obtained in a training set of the particular images.

The microbubbles motion analysis methods proposed in this work are specially useful in the assessment of aortic valve regurgitation and tricuspid valve regurgitation since the microbubbles in different problems have similar presence. The feature selection and classifier training methods described in this dissertation can be obviously adapted to other regurgitation assessment applications as well.
Therefore, MIRAS is not only a potentially powerful clinical aid in quantitative MR assessment, but also potentially useful and source of inspiration in other clinical researches and applications of contrast echocardiography.
References


MIRAS for Contrast 2-D Echocardiograms

References


Appendix I: Program Lists

The constants and variables defined in the following are used in the programs:

```c
#define FRAMNUM 2
#define IMAGNUM 5
#define BACKGROUND 0

#define PARANUM 7
#define ADDNUM 1

#define WIN 128
#define WINLEN 256
#define size 60

#define ROW_OFFSET_ST 8*(8)
#define COL_OFFSET_ST 8*(8)
#define ROW_MATCH_ST 1*(8)
#define COL_MATCH_ST 1*(8)
#define HEIGHT_ST 16*(8)
#define WIDTH_ST 16*(8)
#define ROW_OFFSET_DT 0*(8)
#define COL_OFFSET_DT 8*(8)
#define ROW_MATCH_DT 1*(8)
#define COL_MATCH_DT 1*(8)
#define HEIGHT_DT 16*(8)
#define WIDTH_DT 16*(8)
```
float ap0, ap123, ap1, ap23, ap2, ap3;
float pp[4], pp123, pp23;
int row[5], col[5];
int CalNo, HeartNo, FnamLen, ExchanNo = 0;
char text_buf[255];
int draw[2];
long hst[256];
int lut[256];
int line[512];
int data[128][256];
char fcase[size-2], fn[size+5], fcn[6];
float feat1[6], feat2[7], feat3[6];
float pst[PARANUM][IMAGNUM],
pdt[PARANUM/2+1][IMAGNUM];
int scancode, ascicode;
int execute = 0;
char ny;
int ButtonStatus, PointerCol, PointerRow, NumberPresses,
NumberReleases, HorizCount, VertCount, NumButtons;
*fp_proc, *fp_warn, *fopen();
Sequential Image Acquisition and Alignment Programs

/* image acquisition */

void do_Acq(void)
{
    int thd[3];

    char fcn[]="A.CLB";
    CalNo=1;
    HeartNo=0;
    col[0]=row[0]=120;
    do_ShowAcqText();
    is_reset();
    is_select_ilut(4);
    is_select_olut(6);

    LOOP1:
    is_set_sync_source(1);
    is_select_input_frame(0);
    is_select_output_frame(0);
    is_passthru();

    if(HeartNo==0)
        do_SetInitAcqPara(fn,fcn,&CalNo,&HeartNo,&FnamLen);
    if(HeartNo==0)
        if(HeartNo>IMAGNUM)
            goto QUIT1;
        do_SaveClbAcq(fn,fcn,&CalNo,&HeartNo,&FnamLen,thd);
    }
    if(HeartNo!=0 && CalNo<=FRAMNUM){
        if(HeartNo>IMAGNUM)
            goto QUIT1;
        do_SaveImgAcq(fn,fcn,&CalNo,&HeartNo,&FnamLen,thd);
    }
    is_set_active_region(0,0,512,512);
    goto LOOP1;

QUIT1:
    is_set_active_region(0,0,512,512);
    remove("A.CLB");
    remove("B.CLB");
}
/* initial parameters for image acquisition */

void do_SetInitAcqPara(char fn[],char fcn[],int *CalNo,int *HeartNo,int *FnamLen)
{
    if(*CalNo==1){
        if(acq_state==0){
            do_ShowVideoMenu();
        }
        strncpy(fn,fcase,6);
        *FnamLen=strlen(fn);
    }
    if(*CalNo>FRAMNUM){
        *CalNo=1;
        *HeartNo=1; /* symble */
        fcn[0]='A';
    }
    else{
        ny=do_PressYorN();
        if(ny=='N' || ny=='n')
            *HeartNo=IMAGNUM+1; /* HeartNo > IMAGNUM, exit */
    }
}

/* Acquisition of contrast-free images, acquisition of calibration masks and save in "A.CLB" and "B.CLB" */

void do_SaveClbAcq(char fn[],char fcn[],int *CalNo,int *HeartNo,int *FnamLen,int thd[])
{
    int i,xecg,dif;
    is_acquire(0,1);
    is_set_sync_source(0);
    if(acq_state=='0'){
        do_ecg(&xecg,&dif);
        do_ecg_scale(&xecg);
    }
    else{
        is_and_constant(2,3);
        is_multiply_constant(2,255);
    }
    is_or_frames(0,2);
    is_frame_copy(0,1);
    is_frame_copy(2,0);
    is_cursor(1);
    for(i=0;i<512;++i)
    {
        line[i]=2;
        do_MouReset(&NumButtons);
        scancode=0x00;
    }
}
---*CalNo;
do{
    do_MouGetMickey(&HorizCount,&VertCount);
    if(VertCount || HorizCount){
        col[*CalNo] += HorizCount;
        row[*CalNo] += VertCount;
        if(col[*CalNo] >= 245) col[*CalNo] = 245;
        if(col[*CalNo] <= 10) col[*CalNo] = 10;
        if(row[*CalNo] >= 245) row[*CalNo] = 245;
        if(row[*CalNo] <= 10) row[*CalNo] = 10;
    }
    if(kbint()==1){
        do_PressAcqKey(&row[*CalNo],&col[*CalNo],&scancode,
        &ascicode);
        if(scancode==0x52 && ascicode==0x00){
            is_frame_copy(2,0);
            do_WriteBlock(0,col[*CalNo],row[*CalNo],WINLEN,1,
            line);
            do_WriteBlock(0, col[*CalNo]+255, row[*CalNo], 1,
            WINLEN, line);
            do_WriteBlock(0, col[*CalNo], row[*CalNo]+255,
            WINLEN, 1, line);
            do_WriteBlock(0, col[*CalNo], row[*CalNo], 1,
            WINLEN, line);
        }
    }
    is_set_cursor_position(row[*CalNo]+WIN,col[*CalNo]+WIN);
    do_MouGetButtonPress(0,&ButtonStatus,&NumberPresses,
    &PointerCol,&PointerRow);
    if(NumberPresses==1){ /* if left button press once */
        is_frame_copy(2,0);
        do_WriteBlock(0,col[*CalNo],row[*CalNo],WINLEN,1,line);
        do_WriteBlock(0, col[*CalNo]+255, row[*CalNo], 1,WINLEN,
        line);
        do_WriteBlock(0, col[*CalNo], row[*CalNo]+255,WINLEN,1,
        line);
        do_WriteBlock(0, col[*CalNo], row[*CalNo], 1,WINLEN,line);
    }
    if(ButtonStatus==3){ /* if both button press */
        is_frame_copy(2,0);
    }while(ButtonStatus != 3 && (scancode!=0x1c || ascicode!=0x0d));
    /* when both button or <RET> key is pressed, exit */
    is_cursor(0);
    is_set_active_region(row[*CalNo],col[*CalNo],WINLEN,WINLEN);
    thd[*CalNo]=do_FindMyocardThd(1);
    is_set_active_region(0,0,512,512);
    is_offset_constant(1,-thd[*CalNo]);
    is_multiply_constant(1,300);
    is_and_constant(1,2);
    is_set_active_region(row[*CalNo],col[*CalNo],WINLEN,WINLEN);
    is_copy_region(1,0,row[*CalNo],col[*CalNo]);
    is_set_active_region(0,0,512,512);
    while(1){
        is_set_active_region(row[*CalNo],col[*CalNo],WINLEN,WINLEN);
MIRAS for Contrast 2-D Echocardiograms

is_copy_region(1,0,row[*CalNo],col[*CalNo]);
do_PressAcqKey(&row[*CalNo],&col[*CalNo],&scancode,&ascicode);
if(scancode==0x1c && ascicode==0x0d) /* RET */
  break;
is_frame_copy(2,0);
}
++*CalNo;
is_save(1,0,1,0,fcn);
fn[*FnamLen]=fcn[0];
fn[*FnamLen+1]=0;
fn[*FnamLen+2]=\0;
do_AddNameExtension(fn,"IMG");
is_save(2,0,1,0,fn);
++fcn[0];
++*CalNo;
if(*CalNo>FRAMNUM)
  is_frame_copy(2,1);

/* Acquisition of contrast images */

void do_SaveImgAcq(char fn[],char fcn[],int *CalNo,int *HeartNo,int *FnamLen,int thd[])
{
  int i,ecg,ecg1,ecg2; /* ECG feature */

  is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_restore(2,WIN,WIN,fcn);
--*CalNo;
is_copy_region(2,0,row[*CalNo],col[*CalNo]);
do_ecg_scale(&ecg);
do_ShowUp0();
if(acq_state==00)
  {init_video_control();
   press(SLOW);
   while(1)
     {do_ecg(&ecg1,&ecg2);
       if(abs(ecg2)>0.4*abs(ecg1) && ecg1<xecg+3 && ecg1>xecg-3){
         press(PAUSE);
         _outtext("a");
         do{
           ny=getch();
         } while (ny != ';' && ny !='015');
         if(ny =='015')
           _outtext("a
OK");
         else
           press(SLOW);
       }
       if(ny =='015')
         break;
     }
  }
}
close_video_control();
}
else{
    do{
        inregs.h.ah = 0x00;
        int86(0x16, &inregs, &outregs); /* wait for a key */
        scancode=outregs.h.ah;
        ascicode=outregs.h.al;
    }while (scancode !=0x52 && & ascicode !=0x00  /* Ins */
            && & scancode !=0x1c && & ascicode !=0x0d  /* RET */
            && & scancode !=0x01 && & ascicode !=0x1b);  /* Esc */
}
outtext("a");
if(scancode==0x01 && & ascicode==0x1b) /* Esc */
go_to QUIT3;
if(scancode==0x1c && & ascicode==0x0d) /* RET */
go_to QUIT3;
if(scancode==0x52 && & ascicode==0x00)  /* Ins */
do_imageAlign(&thd[*CalNo],&row[*CalNo],&col[*CalNo],*CalNo);
}
while(1)
    do_pressAcqKey(&row[*CalNo],&col[*CalNo],&scancode,&ascicode);
if((scancode==0x1c && & ascicode==0x0d) || (scancode==0x01 && & ascicode==0x1b))
    break;
outpw(0x230,0xfffc); /* INCSR1 1111 1111 1111 1111 1100 enstop=1 */
outpw(0x230,0xff7c); /* INCSR1 1111 1111 1111 1111 1100 busy=0 */
outpw(0x232,0xff10); /* INCSR2 1111 1111 0001 0000 unprotect bits */
is_set_active_region(row[*CalNo]-10,col[*CalNo]-10,276,276);
is_and_constant(0,252); /* clear old graph */
is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_copy_region(2,0,row[*CalNo],col[*CalNo]);
outpw(0x232,0xff13); /* INCSR2 1111 1111 0001 0011 protect bits */
outpw(0x230,0xff4); /* INCSR1 1111 1111 1111 0100 enstop=0 */
}
QUIT3:
outpw(0x230,0xfffc); /* INCSR1 1111 1111 1111 1111 1100 enstop=1 */
outpw(0x230,0xff7c); /* INCSR1 1111 1111 1111 1111 1100 busy=0 */
outpw(0x232,0xff10); /* INCSR2 1111 1111 0001 0000 unprotect bits */
is_set_active_region(row[*CalNo],col[*CalNo],WINLEN,WINLEN);
is_and_constant(0,252); /* clear old graph */
if(scancode==0x01 && & ascicode==0x1b){
    *HeartNo=IMAGNUM+1; /* HeartNo > IMAGNUM, exit */
go_to QUIT11;
}
fn[*FnamLen]=fcn[0];
if(*HeartNo>9)
    fn[*FnamLen+1]=55+*HeartNo; /* from 'A' */
else
    fn[*FnamLen+1]=48+*HeartNo; /* start from '0' */
fn[*FnamLen+2]=

do_addNameExtension(fn,"IMG");
is_set_active_region(0,0,512,512);
is_and_constant(1,3);
is_multiply_constant(1,255);
is_or_frames(1,0);
is_set_active_region(row[*CalNo],col[*CalNo],WINLEN,WINLEN);
++CalNo;
is_save(0,0,1,0,fn);
is_set_sync_source(1);
++fcn[0];
++*CalNo;
if(*CalNo>FRAMNUM){
  ++*HeartNo;
  *CalNo=1;
  fcn[0]='A';
}
QUIT11:;
}

/* find threshold between cavity and myocardium */

int do_FindMyocardThd(int fb)
{
  int i,tmin,tmax,tn,tm,hmax,thd;

  is_histogram(fb,hst);
  for(i=4;i<256;i+=4)
    if(hst[i]!=0)
      break;
  tmin=i;  /* lowest intensity */
  for(i=252;i>0;i/=4)
    if(hst[i]!=0)
      break;
  tmax=i;  /* highest intensity */
  hmax=0;
  for(i=tmin;i<=(tmin+tmax)/2;i+=4)
    if(hst[i]>hmax){
      hmax=hst[i];  /* maximum histogram in dark region */
      tn=i;
      }
  hmax=0;
  for(i=252;i>=(tmin+tmax)/2;i-=4)
    if(hst[i]>hmax){
      hmax=hst[i];  /* maximum histogram in bright region */
      tm=i;
      }
  thd=(tn+tm)/2;
  return(thd);  /* return threshold value */
}

/* display image through image board from video */
void do_ShowUp(void)
{
    outpw(0x234,0xff26); /* OUTCSR 1111 1111 0010 0110 display off */
    outpw(0x230,0xfffc); /* INCSR1 1111 1111 1111 1100 enstop=1 */
    outpw(0x230,0xff7c); /* INCSR1 1111 1111 0111 1100 busy=0 */
    outpw(0x232,0xff13); /* INCSR2 1111 1111 0001 0011 init, protect 0-1 bits */
    outpw(0x234,0xffa6); /* OUTCSR 1111 1111 1010 0000 display on */
    outpw(0x230,0xff4); /* INCSR1 1111 1111 1111 0100 enstop=0 */
}

/* draw ECG scale */

void do_ecg_scale(int *d)
{
    static int b[2];
    is_set_foreground(1); /* foreground is red */
    is_set_graphic_position(56,200+*d);
    b[0]=86;
    b[1]=200+*d;
    is_draw_lines(0,1,b);
}

/* mask image sector, put outside pixels into intensity 255; original image should be in FB0. FB1 is used for processing, output image in fbo can be either 0 or 2 */

void do_MaskImageSector(void)
{
    int i;
    int xc,yc,x,y,sector23r;
    float RADIUS=270.0;
    sector23r=(sector2r+sector3r)/2;
    is_set_constant(2,255); /* set fb2 with 255 */
    xc=sector1c;
    if(sector1r<256) /* middle vertex is at the up position */
        yc=sector23r-sqrt(RADIUS*RADIUS-(float)(sector1c-sector2c)*(float)
(sector1c-sector2c));
    else /* middle vertex is at the low position */
        yc=sector23r+sqrt(RADIUS*RADIUS-(float)(sector1c-sector2c)*(float)
(sector1c-sector2c));
    for(i=0;i<512;++i)
        line[i]=0; /* black */
    if(sector1r<256){
        for(y=sector1r;y<sector23r;++y)
            }
\[ x = (\text{\texttt{float}})(\text{\texttt{y-sector23r}})* (\text{\texttt{float}})(\text{\texttt{sector1c-sector2c}})/(\text{\texttt{float}})(\text{\texttt{sector1r-sector23r}}) + (\text{\texttt{float}})\text{\texttt{sector2c}}; \]
\[ \text{\texttt{is_put_pixel}}(2,\text{\texttt{y}},\text{\texttt{x}},2*(\text{\texttt{xc-x}})+1,\text{\texttt{line}}); \]
\[ \text{\texttt{for}}(y=\text{\texttt{sector23r}}; y<=\text{\texttt{yc}}+(\text{\texttt{int}})\text{\texttt{RADIUS}}; y++) \{ \]
\[ x = (\text{\texttt{int}})(\text{\texttt{xc}}-\text{\texttt{sqrt}}(\text{\texttt{RADIUS}}^2+\text{\texttt{RADIUS}}-(\text{\texttt{float}})(\text{\texttt{y}}-\text{\texttt{yc}})^2)+0.5); \]
\[ \text{\texttt{is_put_pixel}}(2,\text{\texttt{y}},\text{\texttt{x}},2*(\text{\texttt{xc-x}})+1,\text{\texttt{line}}); \]
\[ \} \]
\[ \text{\texttt{else}} \{ \]
\[ \text{\texttt{for}}(y=\text{\texttt{sector1r}}; y>\text{\texttt{sector23r}}; y-\text{\texttt{y}}) \{ \]
\[ x = (\text{\texttt{float}})(\text{\texttt{y-sector23r}})* (\text{\texttt{float}})(\text{\texttt{sector1c-sector2c}})/(\text{\texttt{float}})(\text{\texttt{sector1r-sector23r}}) + (\text{\texttt{float}})\text{\texttt{sector2c}}; \]
\[ \text{\texttt{is_put_pixel}}(2,\text{\texttt{y}},\text{\texttt{x}},2*(\text{\texttt{xc-x}})+1,\text{\texttt{line}}); \]
\[ \} \]
\[ \text{\texttt{for}}(y=\text{\texttt{sector23r}}; y>=\text{\texttt{yc}}-(\text{\texttt{int}})\text{\texttt{RADIUS}}; y-) \{ \]
\[ x = (\text{\texttt{int}})(\text{\texttt{xc}}-\text{\texttt{sqrt}}(\text{\texttt{RADIUS}}^2+\text{\texttt{RADIUS}}-(\text{\texttt{float}})(\text{\texttt{y}}-\text{\texttt{yc}})^2)+0.5); \]
\[ \text{\texttt{is_put_pixel}}(2,\text{\texttt{y}},\text{\texttt{x}},2*(\text{\texttt{xc-x}})+1,\text{\texttt{line}}); \]
\[ \} \]

\[ \text{\texttt{elseif}} \{ \]
\[ \text{\texttt{for}}(y=\text{\texttt{sector1r}}; y>\text{\texttt{sector23r}}; y-\text{\texttt{y}}) \{ \]
\[ x = (\text{\texttt{float}})(\text{\texttt{y-sector23r}})* (\text{\texttt{float}})(\text{\texttt{sector1c-sector2c}})/(\text{\texttt{float}})(\text{\texttt{sector1r-sector23r}}) + (\text{\texttt{float}})\text{\texttt{sector2c}}; \]
\[ \text{\texttt{is_put_pixel}}(2,\text{\texttt{y}},\text{\texttt{x}},2*(\text{\texttt{xc-x}})+1,\text{\texttt{line}}); \]
\[ \} \]
\[ \text{\texttt{for}}(y=\text{\texttt{sector23r}}; y>=\text{\texttt{yc}}-(\text{\texttt{int}})\text{\texttt{RADIUS}}; y-) \{ \]
\[ x = (\text{\texttt{int}})(\text{\texttt{xc}}-\text{\texttt{sqrt}}(\text{\texttt{RADIUS}}^2+\text{\texttt{RADIUS}}-(\text{\texttt{float}})(\text{\texttt{y}}-\text{\texttt{yc}})^2)+0.5); \]
\[ \text{\texttt{is_put_pixel}}(2,\text{\texttt{y}},\text{\texttt{x}},2*(\text{\texttt{xc-x}})+1,\text{\texttt{line}}); \]
\[ \} \]
\[ \]  

\[ \text{\texttt{/* image alignment */} \] 

\[ \text{\texttt{static char far img[128][512],mask_st[64][256],mask_dt[64][256];} \] 

\[ \text{\texttt{void do_ImageAlign(int *thd, int *row, int *col, int sdt)}} \] 
\[ \text{\texttt{/* thd: threshold for binary image from image */}} \] 
\[ \text{\texttt{/* row, col: upper-left corner coordinates */}} \] 
\[ \text{\texttt{/* imgnum: heart beat image number */}} \] 
\[ \text{\texttt{/* sdt: 0 for systole, 1 for diastole */}} \] 

\[ \text{\texttt{int i,j,k,i,jhmin,jmin,num_st}=0,num_dt=0;}} \] 
\[ \text{\texttt{long s,smin}=600000;}} \] 
\[ \text{\texttt{char imag;}} \] 
\[ \text{\texttt{int row_offset,col_offset,row_match,col_match, height,width;}} \] 
\[ \text{\texttt{is_pass thru();}} \] 
\[ \text{\texttt{is_acquire(0,1);}} \] 
\[ \text{\texttt{is_set_sync_source(0);}} \] 
\[ \text{\texttt{is_load_mask(0);}} \] 
\[ \text{\texttt{is_set_active_region(0,0,512,512);}} \] 
\[ \text{\texttt{if(sdt==0)} \text{\texttt{/* systole image */}} \] 
\[ \text{\texttt{row_offset=ROW_OFFSET_ST;}} \] 
\[ \text{\texttt{col_offset=COL_OFFSET_ST;}} \] 
\[ \text{\texttt{row_match=ROW_MATCH_ST;}} \] 
\[ \text{\texttt{col_match=COL_MATCH_ST;}} \] 
\[ \text{\texttt{height=HEIGHT_ST;}} \] 
\[ \text{\texttt{width=WIDTH_ST;}} \] 
\[ \text{\texttt{)}} \] 
\[ \text{\texttt{if(sdt==1)} \text{\texttt{/* diastole image */}} \] 
\[ \text{\texttt{row_offset=ROW_OFFSET_DT;}} \] 

A 10
col_offset=COL_OFFSET_DT;
row_match=ROW_MATCH_DT;
col_match=COL_MATCH_DT;
height=HEIGHT_DT;
width=WIDTH_DT;
}
for(i=*row+row_offset-row_match;i<*row+row_offset+height+row_match;++i){
    is_get_pixel(0,i,*col+col_offset-col_match,2*col_match+width,line);
    for(k=*col+col_offset-col_match;k<*col+col_offset+width+col_match;++k){
        if(line[k-(*col+col_offset-col_match)] > *thd)
            line[k-(*col+col_offset-col_match)]=1;
        else
            line[k-(*col+col_offset-col_match)]=0;
            img[i/4][k]=img[i/4][k]<<1;
            img[i/4][k]=img[i/4][k] | line[k-(*col+col_offset-col_match)];
    }
}
if(sdt==0 && num_st==0){ /* systole image */
    row_offset=ROW_OFFSET_ST;
    col_offset=COL_OFFSET_ST;
    row_match=ROW_MATCH_ST;
    col_match=COL_MATCH_ST;
    height=HEIGHT_ST;
    width=WIDTH_ST;
    for(i=0;i<height;++i){
        is_get_pixel(2,WIN+row_offset+i,WIN+col_offset,width,line);
        for(k=0;k<width;++k){
            line[k]=line[k]>>1;
            mask_st[i/4][k]=mask_st[i/4][k]<<1;
            mask_st[i/4][k]=mask_st[i/4][k] | line[k];
        }
    }
    ++num_st;
}
if(sdt==1 && num_dt==0){ /* diastole image */
    row_offset=ROW_OFFSET_DT;
    col_offset=COL_OFFSET_DT;
    row_match=ROW_MATCH_DT;
    col_match=COL_MATCH_DT;
    height=HEIGHT_DT;
    width=WIDTH_DT;
    for(i=0;i<height;++i){
        is_get_pixel(2,WIN+row_offset+i,WIN+col_offset,width,line);
        for(k=0;k<width;++k){
            line[k]=line[k]>>1;
            mask_dt[i/4][k]=mask_dt[i/4][k]<<1;
            mask_dt[i/4][k]=mask_dt[i/4][k] | line[k];
        }
    }
    ++num_dt;
}
if(sdt==0){ /* systole image */
    row_offset=ROW_OFFSET_ST;
    col_offset=COL_OFFSET_ST;
    row_match=ROW_MATCH_ST;
    col_match=COL_MATCH_ST;
    height=HEIGHT_ST;
    width=WIDTH_ST;
    imin=jmin=0;

    /* sum of absolute differences in mask searching area */

    for(jj=row_match; jj<=row_match; jj+=8)
        for(j=col_match; j<=col_match; j+=8){
            s=0;
            for(i=row+row_offset-row_match; i<=row+row_offset
                +height+row_match; i+=8)
                for(k=col+col_offset-col_match; k<=col+
                    col_offset+width+col_match; ++k){
                    if(k>=col+j+col_offset && k<col+j+
                        col_offset+width && i>=row+jj+row_offset &&
                        i<row+jj+row_offset+height)
                      imag=imag[i/4][k] ^ mask_st[(i-
                        *row-jj-row_offset)/4][k-*col-j-col_offset];
                    else
                      imag=imag[i/4][k];
                      s+=imag & 0x01;
                      s+=imag & 0x02;
                      s+=imag & 0x04;
                      s+=imag & 0x08;
                      if(k>=col+j+col_offset && k<col+j+
                          col_offset+width && i>=row+jj+row_offset &&
                          i<row+jj+row_offset+height)
                        imag=imag[i/4+1][k] ^ mask_st[(i-
                          *row-jj-row_offset)/4+1][k-*col-j-col_offset];
                    else
                      imag=imag[i/4+1][k];
                      s+=imag & 0x01;
                      s+=imag & 0x02;
                      s+=imag & 0x04;
                      s+=imag & 0x08;

            }

            if(smin>=s){
                smin=s;
                imin=jj;
                jmin=j;
            }

        }

    *row= *row+imin;
    *col=*col+jmin;
    imin=jmin=0;

    /* sum of absolute differences at each 4*4 grid around matched point */

    for(jj=-4; jj<=4; jj+=4)
        for(j=-4; j<=4; j+=4)
            if(jj!=0 || j!=0){

    A . 12 .
s=0;
    for(i=*row+row_offset-4; i<=*row+row_offset+
        height+4; i+=4)
        for(k=*col+col_offset-4; k<=*col+
            col_offset+width+4; ++k){
            if(k>=*col+j+col_offset && k<col
                +j+col_offset+width && i>=*row+jj+row_offset
                && i<row+jj+row_offset+height)
                imag=img[i/4][k] ^ mask_st
                [(i-*row-jj-row_offset)/4][k-*col-j-col_offset];
            else
                imag=img[i/4][k];
                s+=imag & 0x01;
                s+=imag & 0x02;
                s+=imag & 0x04;
                s+=imag & 0x08;
        }
        if(smin>=s){
            smin=s;
            imin=jj;
            jmin=j;
        }
    }
    *row=*row+imin;
    *col=*col+jmin;
}

if(sd==1) {  /* diaastole image */
    row_offset=ROW_OFFSET_DT;
    col_offset=COL_OFFSET_DT;
    row_match=ROW_MATCH_DT;
    col_match=COL_MATCH_DT;
    height=HEIGHT_DT;
    width=WIDTH_DT;
    imin=jmin=0;

    /* sum of absolute differences in mask searching area */
    for(jj=row_match; jj<=row_match; jj+=8)
        for(j=col_match; j<=col_match; j+=8){
            s=0;
            for(i=*row+row_offset-row_match;
                i<=*row+row_offset+height+row_match; i+=8)
                for(k=*col+col_offset-col_match;
                    k<=*col+col_offset+width+col_match; ++k){
                    if(k>=*col+j+col_offset &&
                        k<col+j+col_offset+width && i>=*row+jj+row_offset
                        && i<row+jj+row_offset+height)
                        imag=img[i/4][k] ^ mask_dt[(i-
                        *row-jj-row_offset)/4][k-*col-j-col_offset];
                    else
                        imag=img[i/4][k];
                        s+=imag & 0x01;
                        s+=imag & 0x02;
                        s+=imag & 0x04;
                        s+=imag & 0x08;
if(k>=col+j+col_offset &&
 k<col+j+col_offset+width &&
 i=row+jj+row_offset &&
 i<row+jj+row_offset+height)
 imag=img[i/4][k] ^ mask_dt[i-row-jj-row_offset/4+1][k-*col-j-col_offset];
 else
 imag=img[i/4][k];
 s+=imag & 0x01;
 s+=imag & 0x02;
 s+=imag & 0x04;
 s+=imag & 0x08;

 if(smin>=s) {
 smin=s;
 imin=jj;
 jmin=j;
 }
 */ sum of absolute differences at each 4*4 grid around matched point */

 for(jj=-4;jj<=4;jj+=4)
 for(j=-4;j<=4;j+=4)
 if(jj!=0 || j!=0) {
 s=0;
 for(i=row+row_offset; i<=row+row_offset+
 height+4; i+=4)
 for(k=col+col_offset; k<=col+
 col_offset+width+4; ++k) {

 if(k>=col+j+col_offset &&
 k<col+j+col_offset+width &&
 i=row+jj+row_offset &&
 i<row+jj+row_offset+height)
 [(i-row-jj-row_offset)/4][k-*col-j-col_offset];
 else
 imag=img[i/4][k];
 s+=imag & 0x01;
 s+=imag & 0x02;
 s+=imag & 0x04;
 s+=imag & 0x08;

 if(smin>=s) {
 smin=s;
 imin=jj;
 jmin=j;
 }

 *row+=row+imin;
 *col+=col+jmin;

 is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_set_sync_source(1); /* external */
is_passthru();
is_copy_region(2,0,*row,*col);
outpw(0x232,0xff13); /* INCSR2 1111 1111 0001 0011 protect bits */
outpw(0x230,0xff4);  /* INCSR1 1111 1111 1111 0100 enstop=0 */
printf("\n");
}
ROI Definition Programs

/* ROI definition */

void do_RoiDefine(void)
{
    int row,col,i,nn,thd;
    int fb1=0,fb2=2;

    is_set_sync_source(0);
    do_ShowDefinText();
    strncpy(fn,fcase,6);
    FnamLen=strlen(fn);
    for(CalNo=1;CalNo<=FRAMNUM;++CalNo){
        is_frame_clear(0);
        fn[FnamLen]=64+CalNo;
        fn[FnamLen+1]=48;
        fn[FnamLen+2]=\0;
        do_AddNameExtension(fn,"IMG");
        is_select_orlit(6);
        is_restore(0,WIN,WIN,fn);
        is_and_constant(0,252);
        do_AutoDefinRoi(&thd);

    LOOP2:
        ny=do_RoiDefMenu(fb1,fb2);
        if((ExchanNo/2)*2 != ExchanNo){
            ++ExchanNo;
            is_exchange_frames(0,2);
        }
    switch(ny){
        case 'G':
        case 'g':
            do_CutGraph(fb1,fb2);
            break;
        case 'L':
        case 'l':
            do_RanIncludeGraph(fb1,fb2,WIN,WIN,WIN+WINLEN-1,WIN+WINLEN-1);
            break;
        case 'C':
        case 'c':
            do_CirIncludeGraph(fb1,fb2,WIN,WIN,WIN+WINLEN-1,WIN+WINLEN-1);
            break;
        case 'R':
        case 'r':
            break;
    }
}
do_RectIncludeGraph(fb1, fb2, WIN, WIN, WIN+WINLEN-1, WIN+WINLEN-1);  
  break;
  case 'P':
  case 'p':
    goto QUIT3;
  case '011':
    if(((ExchanNo/2)*2 == ExchanNo){
      ++ExchanNo;
      is_exchange_frames(0,2);
    }
    break;
  }
  goto LOOP2;
QUIT3:
  is_or_frames(2,0);
  do_LabelBlockNo1(2,WIN,WIN,WIN+WINLEN-1,WIN+WINLEN-1);
  for(i=0;i<256;++i)
    lut[i]=0;
  if(CalNo !=2){
    row=col=200; /* up-left position */
    do{
      do_MouseScrollCursor(&row,&col,0,0,511,511);
      do_MouGetButtonPress(0,&ButtonStatus,&NumberPresses,&PointerCol,&PointerRow);
      }while(!NumberPresses); /* left button pressed */
      is_cursor(0);
      printf("\a");
      is_get_pixel(2,row,col,1,&nn);
      lut[nn]=2;
      row=col=300; /* down-right position */
    }else
      row=col=250; /* center position */
    do{
      do_MouseScrollCursor(&row,&col,0,0,511,511);
      do_MouGetButtonPress(0,&ButtonStatus,&NumberPresses,
      &PointerCol,&PointerRow);
      }while(!NumberPresses); /* left button pressed */
      is_cursor(0);
      printf("\a");
      is_get_pixel(2,row,col,1,&nn);
      lut[nn]=1;
      is_load_ilut(4,lut);
      is_perform_feedback(2,4);
  is_and_constant(0,252);
  is_or_frames(2,0);
  is_set_active_region(WIN,WIN,WINLEN,WINLEN);
  for(i=0;i<256;++i)
    lut[i]=i;
  lut[253]=252;
  lut[254]=252;
  lut[255]=252;
/* generate mesh-like masks */

void do_MakeMeshMask(int fb1, int fb2, int row0, int col0)
{
    int i, j;

    is_load_mask(12);
    is_set_active_region(row0, col0, WINLEN/2, WINLEN);
    is_get_region(fb2, data);
    for (i = 0; i < 127; i += 3)
    {
        for (j = 0; j < 253; j += 3)
        {
            data[i][j+1] = 0;
            data[i][j+2] = 0;
        }
    }
    for (j = 0; j < 256; ++j)
    {
        data[i+1][j] = 0;
        data[i+2][j] = 0;
    }
    for (j = 0; j < 256; ++j)
    {
        data[127][j] = 0;
    }
    is_put_region(fb1, data);
    is_set_active_region(row0+WINLEN/2, col0, WINLEN/2, WINLEN);
    is_get_region(fb2, data);
    for (i = 0; i < 127; i += 3)
    {
        for (j = 0; j < 253; j += 3)
        {
            data[i+1][j+1] = 0;
            data[i+1][j+2] = 0;
        }
    }
    for (j = 0; j < 256; ++j)
    {
        data[i][j] = 0;
        data[i+2][j] = 0;
    }
    for (j = 0; j < 253; j += 3)
    {
        data[127][j+1] = 0;
        data[127][j+2] = 0;
    }
    is_put_region(fb1, data);
    is_load_mask(0);
}

MIRAS for Contrast 2-D Echocardiograms
/* ROI definition in contrast-free images */

void do_AutoDefRoi(int *thd)
{
    int i;
    is_frame_copy(0,2);
    do_ReduceImage(0);
    is_and_constant(0,252);
    is_set_active_region(192,192,128,128);
    *thd=do_FindMyocardThd(0);
    for(i=0;i<256;++i)
        lut[i]=0;
    for(i=0;i<*thd;i+=4)
        lut[i]=1;
    is_load_ilut(4,lut);
    is_perform_feedback(0,4);
    is_set_active_region(0,0,512,512);
    do_DilEro5(0,0,1);
    do_DilEro5(0,1,0);
    do_ExpandMask(0);
    is_or_frames(2,0);
    is_frame_copy(0,2);
    is_set_active_region(0,0,512,512);
    for(i=0;i<256;++i)
        lut[i]=2;
    for(i=0;i<256;i+=4)
        lut[i]=0;
    is_load_ilut(4,lut);
    is_perform_feedback(2,4);
    do_MakeMeshMask(0,2,WIN,WIN);
}

/* ROI definition in contrast images */

void do_AutoModifyRoi(char fn[],int FnamLen)
{
    int i,nn,LaLv,thd;
    long hmax;
    for(CalNo=1;CalNo<=FRAMNUM;++CalNo){
        fn[FnamLen]=64+CalNo;
        fn[FnamLen+1]=48;
        fn[FnamLen+2]='0';
        do_AddNameExtension(fn,"IMG");
        is_frame_clear(2);
        is_restore(2,WIN,WIN,fn);
        if(CalNo==1){
            
A. 19.
is_frame_copy(2,1);
for(i=0;i<256;++i)
    lut[i]=0;
for(i=2;i<256;i+=4)
    lut[i]=1;
is_load_lut(4,lut);
is_perform_feedback(1,4);
do_ReduceImage(1);
is_copy_region(1,2,384,0);
do_DilEro7(0,1,0);
do_DilEro7(0,0,1);
is_set_active_region(384,0,128,128);
is_copy_region(2,0,192,192);
is_subtract_frames(1,0);
is_set_active_region(192,192,128,128);
is_copy_region(0,2,384,256);
}
is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_copy_region(2,1,WIN,WIN);
for(i=0;i<256;++i)
    lut[i]=0;
for(i=1;i<256;i+=4)
    lut[i]=1;
is_load_lut(4,lut);
is_perform_feedback(1,4);
do_ReduceImage(1);
is_copy_region(1,2,0,0);
do_DilEro7(0,1,0);
do_DilEro7(0,0,1);
is_set_active_region(0,0,128,128);
is_copy_region(2,0,192,192);
is_subtract_frames(1,0);
is_set_active_region(192,192,128,128);
is_copy_region(0,2,0,256);
for(HeartNo=1;HeartNo<=IMAGNUM;++HeartNo){
    if(HeartNo>9)
        fn[FnamLen+1]=55+HeartNo;
    else
        fn[FnamLen+1]=48+HeartNo;
sprintf(text_buf,"%s",fn);
    outtext(text_buf);
is_restore(1,WIN,WIN,fn);
is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_and_constant(1,252);
is_copy_region(1,2,WIN,WIN);
    LaLv=1;
    if(CalNo==2)
        LaLv=2;
    do{
        if(LaLv==1)
            is_set_active_region(384,256,128,128);
        if(LaLv==2)
            is_set_active_region(0,256,128,128);
        is_copy_region(2,0,192,192);
is_multiply_constant(0,252);
do_ReduceImage(1);
is_and_constant(1,252);
is_and_frames(0,1);

thd=do_FindMyocardThd(1);
for(i=0;i<256;i+=4)
    lut[i]=1;
for(i=0;i<thd;i+=4)
    lut[i]=0;
is_load_ilut(4,lut);
is_perform_feedback(1,4);
if(LwLv==1)
    is_set_active_region(384,0,128,128);
if(LwLv==2)
    is_set_active_region(0,0,128,128);
is_copy_region(2,0,192,192);
is_subtract_frames(1,0);

do_ExpandMask(0);
is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_copy_region(2,1,WIN,WIN);
is_or_frames(1,0);
for(i=0;i<256;++i)
    lut[i]=0;
for(i=1;i<253;i+=4)
    lut[i]=1;
is_load_ilut(4,lut);
is_perform_feedback(0,4);
do_ReduceImage(0);

/* for getting masks of holes */
is_set_constant(1,1);
is_subtract_frames(0,1);
do_LabelBlockNo1(1,192,192,319,319);
is_histogram(1,hst);
hmax=0;
for(i=1;i<256;++i)
    if(hst[i]>hmax){
        hmax=hst[i];
        nn=i;
    }
for(i=1;i<256;++i)
    lut[i]=1;
lut[nn]=0;
is_load_ilut(4,lut);
is_perform_feedback(1,4);
is_or_frames(1,0); /* fill up holes */

do_ExpandMask(0);
is_set_active_region(WIN,WIN,WINLEN,WINLEN);
is_copy_region(2,1,WIN,WIN);
if(CalNo==2)
    is_or_frames(1,0);
else{

A . 21 .
if(LaLv==1){
    is_multiply_constant(0,2);
    is_or_frames(0,1);
    is_copy_region(1,2,WIN,WIN);
}
if(LaLv==2)
    is_or_frames(1,0);
}

while(++LaLv<=2);
    is_save(0,0,1,0,fn);
}
    is_frame_clear(0);

/* 5x5 erosion and dilation operations */

void do_DilEro5(int sign, int fb1, int fb2)
{
    int i,mask[25];
    for(i=0;i<25;++i)
        mask[i]=1;
    if(sign==0){ /* erosion */
        is_convolve(fb1,fb2,5,5,mask,20);
        lut[0]=0;
        lut[1]=1;
        is_load_ilut(4,lut);
        is_perform_feedback(fb2,4);
    }
    if(sign==1){ /* dilation */
        is_convolve(fb1,fb2,5,5,mask,6);
        lut[0]=0;
        for(i=1;i<6;++i)
            lut[i]=1;
        is_load_ilut(4,lut);
        is_perform_feedback(fb2,4);
    }
}

/* 7x7 erosion and dilation operations */

void do_DilEro7(int sign, int fb1, int fb2)
{
    int i, mask[49];
    for(i=0;i<49;++i)
        mask[i]=1;
}

A.22.
if(sign==0) { /* erosion */
    is_convolve(fb1, fb2, 7, 7, mask, 40);
    lut[0]=0;
    lut[1]=1;
    is_load_lut(4, lut);
    is_perform_feedback(fb2, 4);
}

if(sign==1) { /* dilation */
    is_convolve(fb1, fb2, 7, 7, mask, 9);
    lut[0]=0;
    for(i=1;i<7;++i)
        lut[i]=1;
    is_load_lut(4, lut);
    is_perform_feedback(fb2, 4);
}

/* image size reduction */

void do_ResizeImage(int fb1)
{
    int x,y,i;
    for(i=0;i<2;++i){
        for(y=0;y<128;++y)
            do_ReadBlock(fb1, WIN, WIN+128*i+y, WINLEN, 1, data[y]);
        for(y=0;y<128;++y)
            do_WriteBlock(fb1, 192, 192*64+y, 128, 1, data[y]);
    }
    is_set_active_region(WIN+128*1, WIN, 128, WINLEN);
    is_and_constant(fb1, 0);
}

/* enlarge the reduced binary image */

void do_ExpandMask(int fb1)
{
    int x,y,i;
    for(i=0;i<2;++i){
        for(y=0;y<128;++y)
            do_ReadBlock(fb1, 192, 192+64*i+y/2, 128, 1, data[y]);
        data[y][x]=data[(y/2)*2][x/2];
    }
for(y=0;y<128;++y)
    do_WriteBlock(fb1,WIN,WIN+128*i+y,WINLEN,1,data[y]);
}

/* binary object labeling */

void do_LabelBlockNo1(int fb, int x1, int y1, int x2, int y2)
{
    int b[520],r[520],pxe,x,y,n,num,netiq,*et;
    et = malloc(3000*sizeof(int));
    is_set_active_region(x1,y1,x2-x1+1,y2-y1+1);
    for(x=x1-1;x<=x2+1;++x)
        b[x+1]=0;
    netiq=0;
    for(y=y1;y<y2+1;++y){
        pxe=0;
        is_get_pixel(fb,y,x1,x2-x1+1,&r[x1+1]);
        for(x=x1+1;x<=x2+1;++x)
            if((r[x]!=0){
                if(b[x]==0)
                    if(pxe==0)
                        if(r[x+1]==0){
                            ++netiq;
                            et[netiq]=netiq;
                            r[x]=netiq;
                        }
                        else
                            if(b[x+1]!=0)
                                r[x]=b[x+1];
                            else{
                                ++netiq;
                                et[netiq]=netiq;
                                r[x]=netiq;
                            }
                    
                else
                    r[x]=pxe;
            }
            else{
                r[x]=b[x];
                if(r[x]!=0 && pxe!=0)
                    if(et[r[x]]<et[pxe])
                        et[pxe]=et[r[x]];
                    else
                        et[r[x]]=et[pxe];
            }
        pxe=r[x];
        b[x]=r[x];
    }
    else{
        pxe=0;
    }
}
b[x]=r[x];
} is_put_pixel(fb,y,x1,x2-x1+1,&r[x1+1]);
}
for(x=1;x<netiq+1;++x)
    while(et[x]!=et[et[x]])
        et[x]=et[et[x]];
num=0;
for(x=1;x<netiq+1;++x)
    if(x==et[x]){
        ++num;
        for(y=1;y<netiq+1;++y)
            if(et[y]==x)
                et[y]=num;
    }
for(y=y1;y<y2+1;++y){
    is_get_pixel(fb,y,x1,x2-x1+1,&r[x1+1]);
    for(x=x1+1;x<=x2+1;++x){
        n=r[x];
        if(n!=0)
            r[x]=et[n];
    }
    is_put_pixel(fb,y,x1,x2-x1+1,&r[x1+1]);
} free(et);
Features Calculation and Case Classification Programs

/* When ny=='1', calculating parameters from image sequences and saving the
** measurement data to a temporal data file "MIRAS.TMP";
** When ny=='2', reading the measurement data from "MIRAS.TMP"
*/

void do_CalculaPara(void)
{
    int i,j,OptClass,SubOptClass,NewClass;
    int t_alv,t_alal;
    double abv,mbv,anv,mnv,aba,mba,ana,mna;
    long mask_0alv,mask_0ala,sizeLv,sizeLa,h,hmax;
    float para,mask_la,mask_lv,la,lv,area_la,area_lv;

    do{
        ny=getch();
    }while(ny !='1' && ny !='2');
    if(ny=='1'){
        fpw=fopen("MIRAS.TMP","wb"); /* open temporal data file for writing */
        do_GetCaseName(fn);
        fwrite(fn,1,10,fpw); /* write case name */
        FnamLen=strlen(fn);
        fn[FnamLen]=65; /* 'A' frame number digit for image name */
        fn[FnamLen+1]=48; /* start from '0' */
        fn[FnamLen+2]='0';
        do_AddNameExtension(fn,"IMG");
        is_select_out(6);
        is_frame_clear(0);
        is_set_aiactive_region(128,128,256,256);
        for(CalNo=1;CalNo<FRAMNUM;++CalNo){
            fn[FnamLen]=CalNo+64;
            HeartNo=0; /* calculate contrast-free images */
            if(HeartNo>9)
                fn[FnamLen+1]=55+HeartNo; /* from 'A' */
            else
                fn[FnamLen+1]=48+HeartNo; /* start from '0' */
            do_FunctionMenu(19);
            do_InstructionMenu(19);
            do_SystemMenu(19);
            do_InputMenu(19);
            do_ProcessMenu(19);
            sprintf(text_buf,"%s",fn);
            _outtext(text_buf);
            is_restore(0,128,128,fn);
        }
    }
}
is_frame_copy(0,2);
is_frame_copy(2,1);
if(CalNo != 2){
    is_histogram(0,hst);
    abv=0;
    for(i=254;i>=0;i--)
        abv += (i-2)*hst[i];
    for(i=254;i>=0;i--)
        if(hst[i]!=0)
            break;

    mbv=i-2;
    is_and_constant(0,252);
    is_filter(0,1,2);
    is_and_constant(1,252);
    is_subtract_frames(1,0);
    is_frame_copy(2,1);
    is_and_constant(2,2);
    is_histogram(2,hst);
    mask_0alv=hst[2];
    abv /= (double)mask_0alv;
    is_offset_constant(2,2);
    is_load_mask(4);
    is_or_frames(2,0);
    is_load_mask(0);
    is_histogram(0,hst);
    anv=0;
    for(i=252;i>=0;i--)
        anv += i*hst[i];
    anv /= (double)mask_0alv;
    for(i=252;i>=0;i--)
        if(hst[i]!=0)
            break;
}

mnv=i;

t_alv=0.041*(-mbv*0.7+anv*65+mnv*15)+32.6;
/* threshold in LV ROI of ES */
is_frame_copy(1,2);
is_frame_copy(2,0);

}
is_histogram(0,hst);
aba=0;
for(i=253;i>=0;i--)
    aba += (i-1)*hst[i];
for(i=253;i>=0;i--)
    if(hst[i]!=0)
        break;

mba=i-1;
is_and_constant(0,252);
is_filter(0,1,2);
is_and_constant(1,252);
is_subtract_frames(1,0);
is_and_constant(2,1);
is_histogram(2,hst);
mask_0ala=hst[1];
aba/(double)mask_0ala;
is_offset_constant(2,1);
is_load_mask(2);
is_or_frames(2,0);
is_load_mask(0);
is_histogram(0,hst);
anal=0;
for(i=252;i>=0;i--)
    anal+=hst[i];
anal=(double)mask_0ala;
for(i=252;i>=0;i--){
    if(hst[i]!=0)
        break;
}
mna=i;
if(CalNo==1)
    t_al=0.12*(-aba*0.2+0.7*mba+ana*60+mna)+27.4;
    /* threshold in LA ROI of ES */
if(CalNo==2)
    t_al=0.13*(-aba*0.5+mba+ana*55+mna*3)+17;
    /* threshold in LA ROI of ED */

/* calculation from first to last heart beat image */

for(HeartNo=1;HeartNo<=IMAGNUM;++HeartNo){
    if(HeartNo>9)
        fn[FnamLen+1]=55+HeartNo;  /* from 'A' */
    else
        fn[FnamLen+1]=48+HeartNo;  /* start from '0' */
    sprintf(text_buf,"%s",fn);
    _outtext(text_buf);
    _is_restore(0,128,128,fn);
    _is_frame_copy(0,2);
    _is_and_constant(0,252);
    _is_subtract_frames(1,0);
    if(CalNo != 2){
        _is_load_mask(12);
        _is_or_frames(2,0);
        _is_load_mask(0);
        for(i=0;i<256;++i)
            lut[i]=0;
        for(i=2;i<256;i+=4)
            lut[i]=252;
        _is_load_ilut(4,lut);
        _is_perform_feedback(2,4);
        _is_histogram(2,hst);
        mask_lv=hst[252];  /* Alv */
        _is_exchange_frames(0,2);
        _is_and_frames(2,0);
        for(i=0;i<256;++i)
            lut[i]=0;
        for(i=t_alv;i<256;++i)
            lut[i]=252;
        _is_load_ilut(4,lut);
is_perform_feedback(0,4);
is_summation(0,&h);
area_lv=h/252;    /* AMlv */
is_and_frames(2,0);
is_summation(0,&h);
lv=h;            /* IMlv */
is_frame_copy(2,0);
}
for(i=0;i<256;++i)
lut[i]=0;
for(i=1;i<256;i+=4)
lut[i]=252;
is_loadlut(4,lut);
is_perform_feedback(2,4);
is_histogram(2,hst);
mask_la=hst[252];    /* Ala */
is_exchange_frames(0,2);
is_and_frames(2,0);
for(i=0;i<256;++i)
lut[i]=0;
for(i=tala;i<256;++i)
lut[i]=252;
is_loadlut(4,lut);
is_perform_feedback(0,4);
is_summation(0,&h);
area_la=h/252;    /* AMla */
is_and_frames(2,0);
is_summation(0,&h);
la=h;        /* IMla */

if(CalNo != 2){
    pst[0][HeartNo-1]=para=mask_lv;    /* Alv of ES */
    fwrite(&para,4,1,fpw);
    pst[1][HeartNo-1]=para=lv;    /* IMlv of ES */
    fwrite(&para,4,1,fpw);
    pst[2][HeartNo-1]=para=area_lv;    /* AMlv of ES */
    fwrite(&para,4,1,fpw);
    pst[3][HeartNo-1]=para=mask_la;    /* Ala of ES */
    fwrite(&para,4,1,fpw);
    pst[4][HeartNo-1]=para=la;    /* IMla of ES */
    fwrite(&para,4,1,fpw);
    pst[5][HeartNo-1]=para=area_la;    /* AMla of ES */
    fwrite(&para,4,1,fpw);
    if(la*2 != la)    /* if IMla != 0 */
        pst[6][HeartNo-1]=para=

        do_CalculDeviation(128,128,383,383);    /* SDla of ES */
    else
        pst[6][HeartNo-1]=para=0.0;
        fwrite(&para,4,1,fpw);
}
else{
    pdt[0][HeartNo-1]=para=mask_la;    /* Ala of ED */
    fwrite(&para,4,1,fpw);
    pdt[1][HeartNo-1]=para=la;    /* IMla of ED */
A . 29
fwrite(&para,4,1,fpw);
pdt[2][HeartNo-1]=para=area_la; /* AMla of ED */
fwrite(&para,4,1,fpw);
if(la*2 != la) /* if IMla != 0 */
pdt[3][HeartNo-1]=para= /* SDla of ED */
else
pdt[3][HeartNo-1]=para=0.0;
fwrite(&para,4,1,fpw);
}
}
cfclose(fpw);
}
if(ny=='2'){
  fpr=fopen("MIRAS.TMP","rb");
  fread(fn,1,10,fpr);
  for(CalNo=1;CalNo<=FRAMNUM;++CalNo)
    for(HeartNo=1;HeartNo<=IMAGNUM;++HeartNo){
      if(CalNo != 2){
        fread(&para,4,1,fpr);
pst[0][HeartNo-1]=para;
        fread(&para,4,1,fpr);
pst[1][HeartNo-1]=para;
        fread(&para,4,1,fpr);
pst[2][HeartNo-1]=para;
        fread(&para,4,1,fpr);
pst[3][HeartNo-1]=para;
        fread(&para,4,1,fpr);
pst[4][HeartNo-1]=para;
        fread(&para,4,1,fpr);
pst[5][HeartNo-1]=para;
        fread(&para,4,1,fpr);
pst[6][HeartNo-1]=para;
      }
    else{
      fread(&para,4,1,fpr);
pdt[0][HeartNo-1]=para;
      fread(&para,4,1,fpr);
pdt[1][HeartNo-1]=para;
      fread(&para,4,1,fpr);
pdt[2][HeartNo-1]=para;
      fread(&para,4,1,fpr);
pdt[3][HeartNo-1]=para;
    }
  }
}
cfclose(fpr);
}
is_set_active_region(0,0,512,512);
do_Classify(&OptClass,&SubOptClass);
NewClass=do_OutPutValues(OptClass,SubOptClass);
do_JoinNewData(NewClass);
/* calculating standard deviation */

double do_CaloDeviation(int x1, int y1, int x2, int y2)
{
    int x,y,xw,yw;
    long s[50][50],m;
    double xm,ym,dev;

    is_set_active_region(y1,x1,y2-y1+1,x2-x1+1);
    is_summation(0,&m);
    xw=(x2-x1+1)/40;
    yw=(y2-y1+1)/40;
    xm=ym=0;
    for(y=y1;y<y2-yw+1;y+=yw)
        for(x=x1;x<x2-xw+1;x+=xw){
            is_set_active_region(y,x,yw,xw);
            is_summation(0,&s[(y-y1)/yw][(x-x1)/xw]);
            xm+=s[(y-y1)/yw][(x-x1)/xw]*x-x1/xw;
            ym+=s[(y-y1)/yw][(x-x1)/xw]*(y-y1)/yw;
        }
    xm=xm/m;  /* mean position of x */
    ym=ym/m;  /* mean position of y */
    dev=0.0;
    for(y=y1;y<y2;y+=yw)
        for(x=x1;x<x2;x+=xw){
            dev+=((x-x1)/xw-xm)*((x-x1)/xw-xm)+
                  ((y-y1)/yw-ym)*((y-y1)/yw-ym)*s[(y-y1)/yw][(x-x1)/xw];
        }
    is_set_active_region(y1,x1,y2-y1+1,x2-x1+1);
    return(sqrt(dev/(m-1)));  /* standard deviation */
}

/* extract the case name from an input case name sequence, i.e., input [d:] [path] casename in fcase[], output case name in fn[] */

void do_GetCaseName(char fn[])
{
    int i,k=0;

    strncpy(fn,fcase,6);    /* without [d:] and [path] */
    while(fcase[k] !='\0'){
        /* with [d:] or [path] */
        if((fcase[k]=='\n' || fcase[k]=='')
            for(i=0;i<6;++i)
                fn[i]=fcase[i+1+k];
        ++k;
    }
}
/* case classification, output optimal and suboptimal decision classes */

void do_Classify(int *OptClass, int *SubOptClass)
{
  int i,j,k,l[8],m[8];
  float n[6],m00[7],a[50],m11[7],b[50],f[12][IMAGNUM],ref;
  double g[2],p[3];

  /* calculate features from case parameters */

  for(k=0;k<IMAGNUM;++k)
  {
    if(pst[1][k]*2 != pst[1][k])
      f[0][k]=pst[5][k]/pst[2][k];
    else
      f[1][k]=0.0;
    if(pst[5][k]*2 != pst[5][k])
      f[2][k]=pst[6][k];
    else
      f[2][k]=0.0;
    f[3][k]=(pst[4][k]/pst[5][k]+pdt[1][k]/pdt[0][k])/
      (2*pst[1][k]/pdt[0][k]);
    f[4][k]=(pst[5][k]/pst[3][k]+pdt[2][k]/pdt[0][k])/
      (2*pst[2][k]/pdt[0][k]);
    if(pst[5][k]*2 != pst[5][k] && pdt[2][k]*2 != pdt[2][k])
      f[5][k]=((pst[4][k]/pst[5][k]+pdt[1][k]/pdt[2][k])/
        (2*pst[1][k]/pdt[2][k]);
    else
      f[5][k]=0.0;
  }
  else
  {
    f[0][k]=0.0;
    f[1][k]=0.0;
    f[2][k]=0.0;
    f[3][k]=0.0;
    f[4][k]=0.0;
    f[5][k]=0.0;
  }
  f[6][k]=pst[4][k]/pdt[3][k];
  f[7][k]=pst[5][k]/pdt[3][k];
  if(pst[5][k]*2 != pst[5][k])
    f[8][k]=pst[4][k]/pdt[5][k];
  else
    f[8][k]=0.0;
  f[9][k]=pdt[1][k]/pdt[0][k];
  f[10][k]=pdt[2][k]/pdt[0][k];
  if(pdt[2][k]*2 != pdt[2][k])
    f[11][k]=pdt[1][k]/pdt[2][k];
  else
    f[11][k]=0.0;
}
fpr=fopen("MIRAS.LRN","rb");

/* classification at the first node */

feat1[0]=(f[4][2]+f[4][1])/2.0;
feat1[1]=(f[0][0]+f[0][4])/2.0;
feat1[2]=(f[8][4]+f[8][3])/2.0;
feat1[4]=(f[7][2]+f[7][1])/2.0;
feat1[5]=(f[6][2]+f[6][1])/2.0;

/* compute the inverse of the covariance matrix at the first node */

fread(&n[0],4,1,fpr);
for(i=0;i<6;++i)
    fread(&m00[i],4,1,fpr);
    for(j=0;j<6;++j)
        fread(&a[1+i*6+j],4,1,fpr);
}

fread(&n[1],4,1,fpr);
for(i=0;i<6;++i)
    fread(&m11[i],4,1,fpr);
    for(j=0;j<6;++j)
        fread(&b[1+i*6+j],4,1,fpr);
}
for(i=0;i<6;++i)
    for(j=0;j<6;++j)
        a[1+i*6+j]=(a[1+i*6+j]+b[1+i*6+j])/2.0;
do_CalculInverseMatrix(a,l,m,6):

/* compute the discriminant function value of class 0 */

for(i=0;i<2;++i)
    p[i]=0.0;
for(i=0;i<6;++i)
    for(j=0;j<6;++j)
        p[0] +=feat1[i]*a[1+i*6+j]*m00[j];
        p[1] +=m00[i]*a[1+i*6+j]*m00[j];
}
p[2]=log((double)ap0);
g[0]=p[0]-p[1]/2.0+p[2];  /* discriminant function */

/* compute the discriminant function values of pooled class 1,2,3 */

for(i=0;i<2;++i)
    p[i]=0.0;
for(i=0;i<6;++i)
    for(j=0;j<6;++j)
        p[0] +=feat1[i]*a[1+i*6+j]*m11[j];
        p[1] +=m11[i]*a[1+i*6+j]*m11[j];
}
p[2]=log((double)(1.0-ap0));
g[1]=p[0]-p[1]/2.0+p[2];      /* discriminant function */

/* calculate the probability values at the first node */
g[0]=exp(g[0]);
g[1]=exp(g[1]);
pp[0]=g[0]/(g[0]+g[1]);
pp123=1-pp[0];

/* classification at the second node */
feat2[0]=f[3][3]+f[3][2]/2.0;
feat2[1]=(f[4][0]+f[4][4])/2.0;
feat2[2]=f[2][4]-f[2][3];
feat2[3]=(f[3][0]+f[3][4])/2.0;
feat2[4]=(f[1][3]+f[1][2])/2.0;
feat2[5]=f[10][4];
feat2[6]=f[9][4];

/* compute the inverse of the covariance matrix at the second node */
fread(&n[2],4,1,fpr);
for(i=0;i<7;++i){
    fread(&m00[i],4,1,fpr);
    for(j=0;j<7;++j)
        fread(&a[1+i*7+j],4,1,fpr);
}
fread(&n[3],4,1,fpr);
for(i=0;i<7;++i){
    fread(&m11[i],4,1,fpr);
    for(j=0;j<7;++j)
        fread(&b[1+i*7+j],4,1,fpr);
}
for(i=0;i<7;++i){
    for(j=0;j<7;++j)
        a[1+i*7+j]=(a[1+i*7+j]+b[1+i*7+j])/2.0;
}do_CalculInverseMatrix(a,1,m,7);

/* compute the discriminant function value of class 1 */
for(i=0;i<2;++i)
    p[i]=0.0;
for(i=0;i<7;++i){
    for(j=0;j<7;++j)
        p[0]+=feat2[i]*a[1+i*7+j]*m00[j];
        p[1]+=m00[i]*a[1+i*7+j]*m00[j];
}
p[2]=log((double)(ap1/(ap1+ap2+ap3)));
g[0]=p[0]-p[1]/2.0+p[2];      /* discriminant function */

/* compute the discriminant function value of pooled class 2,3 */
for(i=0;i<2;++i)
p[i]=0.0;
for(i=0;i<7;++i)
    for(j=0;j<7;++j){
        p[0] += feat2[i]*a[1+i*7+j]*m11[j];
        p[1] += m11[i]*a[1+i*7+j]*m11[j];
    }
p[2]=log((double)((ap2+ap3)/(ap1+ap2+ap3)));
g[1]=p[0]-p[1]/2.0+p[2]; /* discriminant function */

/* calculate the probability values at the second node */

g[0]=exp(g[0]);
g[1]=exp(g[1]);
pp[1]=g[0]/(g[0]+g[1]);
pp23=1-pp[1];

/* classification at the third node */

feat3[0]=(f[4][3]+f[4][2])/2.0;
feat3[1]=(f[2][2]-f[2][1]);
feat3[2]=f[5][1];
feat3[3]=(f[5][3]-f[5][2]);
feat3[5]=f[1][4]-f[1][3];

/* compute the inverse of the covariance matrix at the third node */

fread(&n[4],4,1,fpr);
for(i=0;i<6;++i)
    fread(&m00[i],4,1,fpr);
    for(j=0;j<6;++j)
        fread(&a[1+i*6+j],4,1,fpr);}

fread(&n[5],4,1,fpr);
for(i=0;i<6;++i)
    fread(&m11[i],4,1,fpr);
    for(j=0;j<6;++j)
        fread(&b[1+i*6+j],4,1,fpr);}

for(i=0;i<6;++i)
    for(j=0;j<6;++j)
        a[1+i*6+j]=(a[1+i*6+j]+b[1+i*6+j])/2.0;

do_CalculInverseMatrix(a,1,m,6);

/* compute the discriminant function value of class 2 */

for(i=0;i<2;++i)
    p[i]=0.0;
for(i=0;i<6;++i)
    for(j=0;j<6;++j){
        p[0] += feat3[i]*a[1+i*6+j]*m00[j];
        p[1] += m00[i]*a[1+i*6+j]*m00[j];
    }
\( p[2] = \log((\text{double})(a2/(a2+a3))) \);
\( g[0] = p[0] - p[1]/2.0 + p[2]; \) /* discriminant function */

/* compute the discriminant function value of class 3 */

for (i=0; i<2; ++i)
   \( p[i] = 0.0; \)
for (i=0; i<6; ++i)
   for (j=0; j<6; ++j)
      \( p[0] += \text{feat3}[i]*a[1+i*6+j]*m11[j]; \)
      \( p[1] += m11[i]*a[1+i*6+j]*m11[j]; \)

\( p[2] = \log((\text{double})(a3/(a2+a3))) \);
\( g[1] = p[0] - p[1]/2.0 + p[2]; \) /* discriminant function */

/* calculate the probability values at the third node */

\( g[0] = \exp(g[0]); \)
\( g[1] = \exp(g[1]); \)
\( pp[2] = g[0]/(g[0]+g[1]); \)

/* calculate probabilities of all classes */

\( pp[0] = pp[0]; \)
\( pp[1] = pp[1]*pp123; \)
\( pp[3] = pp[3]*pp23*pp123; \)

/* find optimal class */

ref = -1;
for (i=0; i<4; ++i)
   if (pp[i] > ref)
      ref = pp[i];
      *OptClass = i;

/* find suboptimal class */

ref = -1;
for (i=0; i<*OptClass; ++i)
   if (pp[i] > ref)
      ref = pp[i];
      *SubOptClass = i;

for (i=*OptClass+1; i<4; ++i)
   if (pp[i] > ref)
      ref = pp[i];
      *SubOptClass = i;

fclose(fpr);
/* subroutine for computing inverse and determinant of a matrix */

float do_CalculInverseMatrix(float a[], int l[], int m[], int n)
{
    int i, ij, ik, iz, ji, jk, jp, jq, jr, k, ki, kj, kk, nk;
    float biga, hold, binv, d;

    d = 1.0;
    nk = -n;
    for(k = 1; k < n + 1; ++k) {
        nk += n;
        l[k] = k;
        m[k] = k;
        kk = nk + k;
        biga = a[kk];
        for(j = k; j < n + 1; ++j) {
            iz = n * (j - 1);
            for(i = k; i < n + 1; ++i) {
                ij = iz + i;
                if(fabs(biga) < fabs(a[ij])) {
                    biga = a[ij];
                    l[k] = i;
                    m[k] = j;
                }
            }
        }
    }

    /* change the rows */
    j = l[k];
    if(i > k) {
        ki = k - n;
        for(i = 1; i < n + 1; ++i) {
            ki += n;
            hold = a[ki];
            ji = ki - k + j;
            a[ki] = a[ji];
            a[ji] = hold;
        }
    }

    /* change the columns */
    i = m[k];
    if(i > k) {
        jp = n * (i - 1);
        for(j = 1; j < n + 1; ++j) {
            jk = nk + j;
            ji = jp + j;
            hold = a[jk];
            a[jk] = a[ji];
            a[ji] = hold;
        }
    }

    /* divide A column for N symmetry of pivot */

A. 37.
/* (the value of pivot is contained in "biga") */
if(biga==0.0F)
{
    d=0.0;
    printf("nSingular Matrix !");
    return(d);
}

binv=1.0/biga;
for(i=1;i<n+1;++i)
    if(i!=k)
    {
        ik=nk+i;
        a[ik]=a[ik]*binv;
    }

/* reduce A matrix */
for(i=1;i<n+1;++i)
    {
        ik=nk+i;
        hold=a[ik];
        ij=i-n;
        for(j=1;j<n+1;++j)
        {
            ij=ij+n;
            if(i!=k)
            {
                if(j!=k)
                {
                    kj=ij-i+k;
                    a[ij]=hold*a[kj]+a[ij];
                }
            }
        }
    }

/* divide A row by pivot */
kj=k-n;
for(j=1;j<n+1;++j)
    {
        kj=kj+n;
        if(j!=k)
        {
            a[kj]=a[kj]*binv;
        }
    }

/* products of pivot */
d *=biga;
/* pivots are replaced by the reverse */
a[kk]=binv;

/* change the last row and column */
k=n;
LOOP1:
do{
    k--;
    if(k>0)
    {
        i=[][k];
        if(i>0)
        {
            jq=n*(k-1);
            jr=n*(i-1);
            for(j=1;j<n+1;++j)
            {
                jk=jq+j;
                hold=a[jk];
                ji=jr+j;
                a[jk]=a[ji];
                a[ji]=hold;
            }
        }
    }
}

A. 38 .
MIRAS for Contrast 2-D Echocardiograms

Appendix I

```c
}
}
j=m[k];
if(j>k){
    ki=k-n;
    for(i=1;i<n+1;++i){
        ki=ki+n;
        hold=a[ki];
        ji=ki-k+j;
        a[ki]=a[ji];
        a[ji]=hold;
    }
}
else
    goto LOOP1;
}
else
    return(d);
}while(1);
}

/* appending temporal measurement data in "MIRAS.TMP" to "MIRAS.DAT" */

void do_JoinNewData(int NewClass)
{
    float para;
    int i,j,k;

    fpr=fopen("MIRAS.TMP","rb"); /* open TEMP data file for reading */
    fpa=fopen("MIRAS.NEW","ab");  /* open NEW data file for appending */
    fread(fn1,10,fpr);
    fwrite(fn1,10,fpa);              /* case name */
    para=NewClass;
    fwrite(&para,4,1,fpa);           /* Class No. */
    for(i=0;i<FRAMNUM;++i){
        if(i != 1)
            for(k=0;k<IMAGNUM+BACKGROUND;++k)
                for(j=0;j<PARANUM;++j){
                    fread(&para,4,1,fpr);
                    fwrite(&para,4,1,fpr);
                }
        else
            for(k=0;k<IMAGNUM+BACKGROUND;++k)
                for(j=0;j<PARANUM/2+1;++j){
                    fread(&para,4,1,fpr);
                    fwrite(&para,4,1,fpr);
                }
    }
    fclose(fpr);
    fclose(fpa);
}```
Training Database Management Programs

/* Generating a database in "MIRAS.LRN" from existing measurement data in
** "MIRAS.DAT"
*/

float para,n[6],s[6][7][7],s0[6][7],[7],c[6][7][7],m0[6][7],m1[6][7],
f[12][IMAGNUM+BACKGROUND],ff[7];

void do_GenerateNewDataBase(void)
{
    int i,j,k;
    for(k=0;k<6;++k){
        n[k]=0.0;
        for(i=0;i<7;++i){
            m0[k][i]=0.0;
            for(j=0;j<7;++j)
                s0[k][i][j]=0.0;
        }
    }

    /* read existing database from "MIRAS.DAT" */

    fpr=fopen("MIRAS.DAT","rb");
    while((ny=getc(fpr)) != EOF){
        for(i=0;i<9;++i)
            ny=getc(fpr);
        fread(&para,4,1,fpr);
        for(i=0;i<FRAMNUM;++i){
            if(i != 1)
                for(k=0;k<IMAGNUM+BACKGROUND;++k)
                    for(j=0;j<PARANUM;++j)
                        fread(&pst[j][k],4,1,fpr);
            else
                for(k=0;k<IMAGNUM+BACKGROUND;++k)
                    for(j=0;j<PARANUM/2+1;++j)
                        fread(&pdt[j][k],4,1,fpr);

            for(k=0;k<IMAGNUM+BACKGROUND;++k){
                if(pst[1][k]*2 != pst[1][k]){
                    f[0][k]=pst[5][k]/pst[2][k];
                    if(pst[5][k]*2 != pst[5][k])
                        f[1][k]=(pst[4][k]/pst[5][k])/pst[1][k]/pst[2][k];
                else
                    f[1][k]=0.0;
            }
        }
    }
}
\[ f[2][k] = p[st][6][k]; \]
\[ f[3][k] = (p[st][4][k]/p[st][3][k]+pdt[1][k]/pdt[0][k]) / (2*pst[1][k]/pdt[0][k]); \]
\[ f[4][k] = (p[st][5][k]/p[st][3][k]+pdt[2][k]/pdt[0][k]) / (2*pst[2][k]/pdt[0][k]); \]
\[ if(p[st][5][k]*2 != pst[5][k] && pdt[2][k]*2 != pdt[2][k]) \]
\[ f[5][k] = (p[st][4][k]/p[st][5][k]+pdt[1][k]/pdt[2][k])/ (2*pst[1][k]/pdt[2][k]); \]
\[ else \]
\[ f[5][k] = 0.0; \]
\[ } \]
\[ else \]
\[ f[0][k] = 0.0; \]
\[ f[1][k] = 0.0; \]
\[ f[2][k] = 0.0; \]
\[ f[3][k] = 0.0; \]
\[ f[4][k] = 0.0; \]
\[ f[5][k] = 0.0; \]
\[ } \]
\[ f[6][k] = pst[4][k]/p[st][3][k]; \]
\[ f[7][k] = pst[5][k]/p[st][3][k]; \]
\[ if(p[st][5][k]*2 != pst[5][k]) \]
\[ f[8][k] = pst[4][k]/p[st][5][k]; \]
\[ else \]
\[ f[8][k] = 0.0; \]
\[ f[9][k] = pdt[1][k]/pdt[0][k]; \]
\[ f[10][k] = pdt[2][k]/pdt[0][k]; \]
\[ if(pdt[2][k]*2 != pdt[2][k]) \]
\[ f[11][k] = pdt[1][k]/pdt[2][k]; \]
\[ else \]
\[ f[11][k] = 0.0; \]
\[ } \]

/* compute means and covariance matrix of every group at each node */

\[ f[0] = (f[4][2]+f[4][1])/2.0; \]
\[ f[1] = (f[0][0]+f[0][4])/2.0; \]
\[ f[2] = (f[8][4]+f[8][3])/2.0; \]
\[ f[4] = (f[7][2]+f[7][1])/2.0; \]
\[ f[5] = (f[6][2]+f[6][1])/2.0; \]
\[ if(para == 0.0) \}
\[ +n[0]; \]
\[ for(k=0;k<6;++k) \}
\[ m1[0][k] = (m[0][0]-1)*m0[0][k]+ff[k]/n[0]; \]
\[ m0[0][k] = m1[0][k]; \]
\[ } \]
\[ for(i=0;i<6;++i) \]
\[ for(j=0;j<6;++j) \}
\[ s[0][i][j] = (m[0][0]-1)*s[0][i][j]+ff[i]*ff[j]/n[0]; \]
\[ s[0][i][j] = s[0][i][j]; \]
\[ } \]
\[ if(para != 0.0) \}
MIRAS for Contrast 2-D Echocardiograms

++n[1];
    for(k=0;k<6;++k){
        m1[1][k]=((n[1]-1)*m0[1][k]+ff[k])/n[1];
        m0[1][k]=m1[1][k];
    }
    for(i=0;i<6;++i)
        for(j=0;j<6;++j){
            s1[1][i][j]=((n[1]-1)*s0[1][i][j]+ff[i]*ff[j])/n[1];
            s0[1][i][j]=s1[1][i][j];
        }

ff[0]=(f[3][3]+f[3][2])/2.0;
ff[1]=(f[4][0]+f[4][1])/2.0;
ff[2]=f[2][4]-f[2][3];
ff[3]=(f[3][0]+f[3][1])/2.0;
ff[4]=(f[1][1]+f[1][2])/2.0;
ff[5]=f[10][4];
ff[6]=f[9][4];
if(para==1.0){
    ++n[2];
    for(k=0;k<7;++k){
        m1[2][k]=((n[2]-1)*m0[2][k]+ff[k])/n[2];
        m0[2][k]=m1[2][k];
    }
    for(i=0;i<7;++i)
        for(j=0;j<7;++j){
            s2[1][i][j]=((n[2]-1)*s0[2][i][j]+ff[i]*ff[j])/n[2];
            s0[2][i][j]=s2[1][i][j];
        }
}
if(para!=1.0 && para!=0.0){
    ++n[3];
    for(k=0;k<7;++k){
        m1[3][k]=((n[3]-1)*m0[3][k]+ff[k])/n[3];
        m0[3][k]=m1[3][k];
    }
    for(i=0;i<7;++i)
        for(j=0;j<7;++j){
            s3[1][i][j]=((n[3]-1)*s0[3][i][j]+ff[i]*ff[j])/n[3];
            s0[3][i][j]=s3[1][i][j];
        }
}
ff[0]=(f[4][3]+f[4][2])/2.0;
ff[1]=f[2][2]-f[2][1];
ff[2]=f[5][1];
ff[3]=f[5][3]-f[5][2];
ff[5]=f[1][4]-f[1][3];
if(para==2.0){
    ++n[4];
    for(k=0;k<6;++k){
        m1[4][k]=((n[4]-1)*m0[4][k]+ff[k])/n[4];
        m0[4][k]=m1[4][k];
    }
}
for (i=0; i<6; ++i)
    for (j=0; j<6; ++j)
        s[4][i][j] = ((n[4]-1)*s0[4][i][j]+ff[i]*ff[j])/n[4];
    s0[4][i][j] = s[4][i][j];
}
if (para==3.0) {
    ++n[5];
    for (k=0; k<6; ++k) {
        m1[5][k] = ((n[5]-1)*m0[5][k]+ff[k])/n[5];
        m0[5][k] = m1[5][k];
    }
    for (i=0; i<6; ++i)
        for (j=0; j<6; ++j) {
            s[5][i][j] = ((n[5]-1)*s0[5][i][j]+ff[i]*ff[j])/n[5];
            s0[5][i][j] = s[5][i][j];
        }
}
fclose(fpr);
/* compute covariance matrix elements */
for (k=0; k<2; ++k)
    for (i=0; i<6; ++i)
        for (j=0; j<6; ++j)
            c[k][i][j] = s[k][i][j] - m0[k][i]*m0[k][j];
for (k=2; k<4; ++k)
    for (i=0; i<7; ++i)
        for (j=0; j<7; ++j)
            c[k][i][j] = s[k][i][j] - m0[k][i]*m0[k][j];
for (k=4; k<6; ++k)
    for (i=0; i<6; ++i)
        for (j=0; j<6; ++j)
            c[k][i][j] = s[k][i][j] - m0[k][i]*m0[k][j];
/* save covariance matrix elements to a data file "MIRAS.LRN" */
fpw=fopen("MIRAS.LRN","wb");
for (k=0; k<2; ++k) {
    fwrite(&n[k],4,1,fpw);
    for (i=0; i<6; ++i) {
        fwrite(&m0[k][i],4,1,fpw);
        for (j=0; j<6; ++j)
            fwrite(&c[k][i][j],4,1,fpw);
    }
}
for (k=2; k<4; ++k) {
    fwrite(&n[k],4,1,fpw);
    for (i=0; i<7; ++i) {
        fwrite(&m0[k][i],4,1,fpw);
        for (j=0; j<7; ++j)
            fwrite(&c[k][i][j],4,1,fpw);
    }
}
for(k=4;k<6;++k){
    fwrite(&n[k],4,1,fpw);
    for(i=0;i<6;++i){
        fwrite(&m0[k][i],4,1,fpw);
        for(j=0;j<6;++j)
            fwrite(&c[k][i][j],4,1,fpw);
    }
}fclose(fpw);

/* Updating the database in "MIRAS.LRN" with the new measurement data in */
/* "MIRAS.NEW", appending the new data "MIRAS.NEW" to the existing */
/* data "MIRAS.DAT" and deleting "MIRAS.NEW" */

void do_UpdateDataBase(void)
{
    int i,j,k;
    float para;

    /* read data from the existing database in "MIRAS.LRN" */
    fpr=fopen("MIRAS.LRN","rb");
    for(k=0;k<2;++k){
        fread(&n[k],4,1,fpr);
        for(i=0;i<6;++i){
            fread(&m0[k][i],4,1,fpr);
            for(j=0;j<6;++j)
                fread(&c[k][i][j],4,1,fpr);
        }
    }
    for(k=2;k<4;++k){
        fread(&n[k],4,1,fpr);
        for(i=0;i<7;++i){
            fread(&m0[k][i],4,1,fpr);
            for(j=0;j<7;++j)
                fread(&c[k][i][j],4,1,fpr);
        }
    }
    for(k=4;k<6;++k){
        fread(&n[k],4,1,fpr);
        for(i=0;i<6;++i){
            fread(&m0[k][i],4,1,fpr);
            for(j=0;j<6;++j)
                fread(&c[k][i][j],4,1,fpr);
        }
    }fclose(fpr);
for(k=0;k<2;++k)
  for(i=0;i<6;++i)
    for(j=0;j<6;++j)
      s[k][i][j]=c[k][i][j]+m0[k][i]*m0[k][j];
for(k=2;k<4;++k)
  for(i=0;i<7;++i)
    for(j=0;j<7;++j)
      s[k][i][j]=c[k][i][j]+m0[k][i]*m0[k][j];
for(k=4;k<6;++k)
  for(i=0;i<6;++i)
    for(j=0;j<6;++j)
      s[k][i][j]=c[k][i][j]+m0[k][i]*m0[k][j];

/* read new measurement data from "MIRAS.NEW", append to "MIRAS.DAT", and calculate features */

fpr=fopen("MIRAS.NEW","rb");
fpa=fopen("MIRAS.DAT","ab");
while((ny=getc(fpr)) != EOF){
  putc(ny,fpa);
  for(i=0;i<9;++i){
    ny=getc(fpr);
    putc(ny,fpa);
  }
  fread(&para,4,1,fpr);
  fwrite(&para,4,1,fpa);
  for(i=0;i<FRAMENUM;++i){
    if(i != 1)
      for(k=0;k<IMAGNUM+BACKGROUND;++k)
        for(j=0;j<PARANUM;++j){
          fread(&pst[j][k],4,1,fpr);
          fwrite(&pst[j][k],4,1,fpa);
        }
    else
      for(k=0;k<IMAGNUM+BACKGROUND;++k)
        for(j=0;j<PARANUM/2+1;++j){
          fread(&pdt[j][k],4,1,fpr);
          fwrite(&pdt[j][k],4,1,fpa);
        }
  }
}
for(k=0;k<IMAGNUM+BACKGROUND;++k){
  if(pst[1][k]*2 != pst[1][k]){
    ff0[k]=pst[5][k]/pst[2][k];
    if(pst[5][k]*2 != pst[5][k])
      f1[k]=(pst[4][k]/pst[5][k])/(pst[1][k]/pst[2][k]);
    else
      f1[k]=0.0;
    f2[k]=pdt[6][k];
    f3[k]=(pdt[4][k]/pdt[3][k]+pdt[1][k]/pdt[0][k])/
(2*pst[1][k]/pdt[0][k]);
    f4[k]=(pdt[5][k]/pdt[3][k]+pdt[2][k]/pdt[0][k])/
(2*pst[2][k]/pdt[0][k]);
    if(pst[5][k]*2 != pst[5][k] &amp; pdt[2][k]*2 != pdt[2][k])
(2*pst[1][k]/pst[2][k]);
    else
        f[5][k]=0.0;
}
else{
    f[0][k]=0.0;
    f[1][k]=0.0;
    f[2][k]=0.0;
    f[3][k]=0.0;
    f[4][k]=0.0;
    f[5][k]=0.0;
}
f[6][k]=pst[4][k]/pst[3][k];
f[7][k]=pst[5][k]/pst[3][k];
if(pst[5][k]*2 !=-pst[5][k])
    f[8][k]=pst[4][k]/pst[5][k];
else
    f[8][k]=0.0;
    f[9][k]=pdt[1][k]/pdt[0][k];
    f[10][k]=pdt[2][k]/pdt[0][k];
    if(pdt[2][k]*2 !=-pdt[2][k])
        f[11][k]=pdt[1][k]/pdt[2][k];
else
    f[11][k]=0.0;
}

/* compute means and covariance matrix of every group at each node */

ff[0]=(f[4][2]+f[4][1])/2.0;
ff[1]=(f[0][0]+f[0][4])/2.0;
ff[2]=(f[8][4]+f[8][3])/2.0;
ff[4]=(f[7][2]+f[7][1])/2.0;
ff[5]=(f[6][2]+f[6][1])/2.0;
if(para==0.0){
    ++n[0];
    for(k=0;k<6;++k) {
        m1[0][k]=((n[0]-1)*m0[0][k]+ff[k])/n[0];
        m0[0][k]=m1[0][k];
    }
    for(i=0;i<6;++i){
        for(j=0;j<6;++j){
            s0[i][j]=(n[0]-1)*s0[i][j]+ff[i]*ff[j])/n[0];
            s0[i][j]=s0[i][j];
        }
    }
}
if(para!=0.0){
    ++n[1];
    for(k=0;k<6;++k) {
        m1[1][k]=((n[1]-1)*m0[1][k]+ff[k])/n[1];
        m0[1][k]=m1[1][k];
    }
}
for(i=0;i<6;++i)
    for(j=0;j<6;++j) {
        s[1][i][j] = ((n[1]-1)*s0[1][i][j]+ff[i]*ff[j])/n[1];
        s0[1][i][j] = s[1][i][j];
    }

ff[0]=(f[3][3]+f[3][2])/2.0;
ff[1]=(f[4][0]+f[4][4])/2.0;
ff[2]=(f[2][4]-f[2][3]);
ff[3]=(f[3][0]+f[3][4])/2.0;
ff[4]=(f[1][3]+f[1][2])/2.0;
ff[5]=f[10][4];
ff[6]=f[9][4];
if(para==1.0) {
    ++n[2];
    for(k=0;k<7;++k) {
        m1[2][k] = ((n[2]-1)*m0[2][k]+ff[k])/n[2];
        m0[2][k] = m1[2][k];
    }
    for(i=0;i<7;++i)
        for(j=0;j<7;++j) {
            s[2][i][j] = ((n[2]-1)*s0[2][i][j]+ff[i]*ff[j])/n[2];
            s0[2][i][j] = s[2][i][j];
        }
}
if(para!=1.0 && para!=0.0) {
    ++n[3];
    for(k=0;k<7;++k) {
        m1[3][k] = ((n[3]-1)*m0[3][k]+ff[k])/n[3];
        m0[3][k] = m1[3][k];
    }
    for(i=0;i<7;++i)
        for(j=0;j<7;++j) {
            s[3][i][j] = ((n[3]-1)*s0[3][i][j]+ff[i]*ff[j])/n[3];
            s0[3][i][j] = s[3][i][j];
        }
}
ff[0]=(f[4][3]+f[4][2])/2.0;
ff[1]=(f[2][2]-f[2][1]);
ff[2]=f[5][1];
ff[3]=f[5][3]-f[5][2];
ff[5]=f[1][4]-f[1][3];
if(para==2.0) {
    ++n[4];
    for(k=0;k<6;++k) {
        m1[4][k] = ((n[4]-1)*m0[4][k]+ff[k])/n[4];
        m0[4][k] = m1[4][k];
    }
    for(i=0;i<6;++i)
        for(j=0;j<6;++j) {
            s[4][i][j] = ((n[4]-1)*s0[4][i][j]+ff[i]*ff[j])/n[4];
            s0[4][i][j] = s[4][i][j];
        }
}

A . 47 .

MIRAS for Contrast 2-D Echocardiograms

Appendix I
} } 

if(para==3.0){
    ++n[5];
    for(k=0;k<6;++k){
        m1[5][k]=((n[5]-1)*m0[5][k]+ff[k])/n[5];
        m0[5][k]=m1[5][k];
    }
    for(i=0;i<6;++i)
        for(j=0;j<6;++j){
            s[5][i][j]=((n[5]-1)*s0[5][i][j]+ff[i]*ff[j])/n[5];
            s0[5][i][j]=s[5][i][j];
        }
} 

/* compute covariance matrix elements */

for(k=0;k<2;++k)
    for(i=0;i<6;++i)
        for(j=0;j<6;++j)
            c[k][i][j]=s[k][i][j]-m0[k][i]*m0[k][j];

for(k=2;k<4;++k)
    for(i=0;i<7;++i)
        for(j=0;j<7;++j)
            c[k][i][j]=s[k][i][j]-m0[k][i]*m0[k][j];

for(k=4;k<6;++k)
    for(i=0;i<6;++i)
        for(j=0;j<6;++j)
            c[k][i][j]=s[k][i][j]-m0[k][i]*m0[k][j];

/* save covariance matrix elements to a data file "MIRAS.LRN" */

fpw=fopen("MIRAS.LRN","wb");
for(k=0;k<2;++k){
    fwrite(&n[k],4,1,fpw);
    for(i=0;i<6;++i){
        fwrite(&m0[k][i],4,1,fpw);
        for(j=0;j<6;++j)
            fwrite(&c[k][i][j],4,1,fpw);
    }
}
for(k=2;k<4;++k){
    fwrite(&n[k],4,1,fpw);
    for(i=0;i<7;++i){
        fwrite(&m0[k][i],4,1,fpw);
        for(j=0;j<7;++j)
            fwrite(&c[k][i][j],4,1,fpw);
    }
}
for(k=4;k<6;++k){
    fwrite(&n[k],4,1,fpw);
    for(i=0;i<6;++i){
        fwrite(&m0[k][i],4,1,fpw);
        for(j=0;j<6;++j)
            fwrite(&c[k][i][j],4,1,fpw);
    }
}

A.48
fwrite(&c[k][i][j],4,1,fpw);

}
fclose(fpw);
fclose(fpr);
fclose(fp);
remove("MIRAS.NEW");

1.1. File Names and Extensions

MIRAS.

Each image is associated with a sequence of file names used for a single study, as follows:

DATABASE DATAX DATABASE DATALOG

DATABASE and DATABASE are sequences of names that identify the study, patient, and study, respectively.

DATABASE and DATABASE are also used to determine the position of the study, patient, and study, respectively.

DATABASE and DATABASE are used to store data and are not affected by the present study.

DATABASE is used to store data and is not affected by the present study.

DATABASE is used to store data and is not affected by the present study.
Appendix II: MIRAS Data and Operation

II.I File Names and Structures:

MIRAS.IDX

This is an index file consisting of case names used for a batch job processing. The data structure is NAME1, NAME2, NAME3, NAME4, .... Each name is composed of 6 characters.

MIRAS.INI

This is an initial parameter file consisting of prevalence of each class, coordinates of the three corners of the image sector, and a sign for auto or manual image acquisition. The data structure is as follows:

DATA1, DATA2, DATA3, ... , DATA9, DATA10, DATA11.

DATA1, DATA2, DATA3, and DATA4 are prevalences of Class 0, Class 1, Class 2, and Class 3, respectively.
DATA5 and DATA6 are row and column coordinates at the sector vertex.
DATA7 and DATA8 are row and column coordinates at the sector left corner.
DATA9 and DATA10 are row and column coordinates at the sector right corner.
DATA11 is a sign. 0 for auto, 1 for manual.

All data are floating point type.
MIRAS.DAT

This is a confirmed database consisting of the case name, authorized class number, and basic parameters values. The data structure is in the following:

STRING, CLASS, DATA1, DATA2, ..., DATA54, DATA55, <RET>

STRING is a case name covering 10 characters.
CLASS is the class number.
DATA1 ~ DATA7 are respectively Alv, IMlv, AMlv, Ala, IMla, AMla, SDla in ES image phases of the first heart beat.
DATA8 ~ DATA14, DATA15 ~ DATA21, DATA22 ~ DATA28, and DATA29 ~ DATA35 are those parameters in ES image phases of, respectively, the second, third, fourth and fifth heart beats.
DATA36 ~ DATA39 are respectively Ala, IMla, AMla, SDla in ED image phase of the first heart beat.
DATA40 ~ DATA43, DATA44 ~ DATA47, DATA48 ~ DATA51, and DATA52 ~ DATA55 are those parameters in ED image phases of, respectively, the second, third, fourth and fifth heart beats.
<RET> is the carriage return character as a sign of the end of a case parameters.

The CLASS and all the DATA are floating point type.

MIRAS.NEW

This is a newly processed data file having the same data structure as that of MIRAS.DAT. After the data in MIRAS.NEW are used for updating the confirmed database MIRAS.DAT by an authorized doctor, MIRAS.NEW is deleted.

MIRAS.TMP

Data before classification being processed from the cases of the MIRAS.IDX file. The data structure is the same as that of MIRAS.DAT when deleting the CLASS parameter. When classification is done, the data in MIRAS.TMP and the class number are appended to MIRAS.NEW, and MIRAS.TMP is then deleted.
This is a classification parameter file obtained from MIRAS.DAT. It consists of the number of cases, mean and covariances in each class, computed from Equations (4.33), (4.34) and (4.35). All data in MIRAS.DAT are floating point type and arranged as follows:

NUM1, MEANA1, COVA1, ..., COVA6, MEANA2, COVA7, ..., MEANA6, COVA31, ..., COVA36, NUM2, MEANB1, COVB1, ..., COVB36, NUM3, MEANC1, COVC1, ..., COVC49, NUM4, MEAND1, COVD1, ..., COVD49, NUM5, MEANE1, COVE1, ..., COVE36, NUM6, MEANF1, COVF1 ..., COVF36.

NUM1, NUM2, NUM3, NUM4, NUM5, and NUM6 are the numbers of cases in Class 0, Class 1,2,3, Class 1, Class 2,3, Class 2, and Class 3, respectively.

MEANA1, ..., MEANA6 are mean values of respective 6 features in Class 0.
MEANB1, ..., MEANB6 are mean values of respective 6 features in Class 1,2,3.
MEANC1, ..., MEANC7 are mean values of respective 7 features in Class 1.
MEAND1, ..., MEAND7 are mean values of respective 7 features in Class 2,3.
MEANE1, ..., MEANE6 are mean values of respective 6 features in Class 2.
MEANF1, ..., MEANF6 are mean values of respective 6 features in Class 3.
COVA1, COVA2, ..., COVA36 are 6x6 covariance matrix of Class 0.
COVB1, COVB2, ..., COVB36 are 6x6 covariance matrix of Class 1,2,3.
COVC1, COVC2, ..., COVC49 are 7x7 covariance matrix of Class 1.
COVD1, COVD2, ..., COVD49 are 7x7 covariance matrix of Class 2,3.
COVE1, COVE2, ..., COVE36 are 6x6 covariance matrix of Class 2.
COVF1, COVF2, ..., COVF36 are 6x6 covariance matrix of Class 3.

MIRAS.EXE: MIRAS main program.

A.CLB: Template for ES image phase.
B.CLB: Template for ED image phase.

*A0.IMG: ES contrast-free image.
*B0.IMG: ED contrast-free image.
*A1.IMG: ES contrast image heart beat number 1.
*B1.IMG: ED contrast image heart beat number 1.
*A2.IMG: ES contrast image heart beat number 2.
*B2.IMG: ED contrast image heart beat number 2.
*A3.IMG: ES contrast image heart beat number 3.
*B3.IMG: ED contrast image heart beat number 3.
*A4.IMG: ES contrast image heart beat number 4.
*B4.IMG: ED contrast image heart beat number 4.
*A5.IMG: ES contrast image heart beat number 5.
*B5.IMG: ED contrast image heart beat number 5.

("*" is a case name).

2. Definition of Regions of Interest.

These regions are labeled A and B on the ECG and are placed over the LV and RV ends of the long axis view. The LV area is delineated by the endocardial border of both the anterior and posterior walls.
II.2 Operation Phases

1. Image Acquisition

Echocardiographic images are acquired by the computer from a VHS digital tape recorder and automatically aligned.

1.1 Define a case name (up to 6 characters) which will be used for identifying the created files. Suggested name: patient abbreviation followed by case number (e.g. MMS004)

1.2 Play the tape, detect the end-systole image corresponding to the preinjection moment and freeze it.

1.3 Define the interest area of the image with the mouse and grab it (return key).

1.4 Repeat the previous steps for the end-diastole image.

1.5 Repeat steps 1.2 and 1.4 for 5 heart beats after contrast agent injection.

1.6 Press <Ins> key for image alignment and press <Enter> key to grab it.

Semi-automatic operation
Approx. execution time: 7 minutes per case

2. Definition of Regions of Interest

ROIs are defined in the LV and LA chambers in ES phases and LA chamber in ED phases of both contrast-free images in a semiautomatic way. ROIs are automatically defined in contrast images.
2.1 Modify ROI masks in ES contrast-free image with mouse and developed tools.

2.2 Repeat step 2.1 for ED phase.

2.3 Point out LV ROI and LA ROI in ES contrast-free image with mouse.

2.4 Repeat step 2.3 for LA ROI in ED phase.

Semi-automatic operation
Approx. execution time: 6 minutes per case

3. Case Classification

MR features of the case are automatically computed and classified. A classification report is given by the computer and waiting for the judgement by an authorized doctor.

3.1 Input authorized classification.

Automatic operation
Approx. execution time: 7 minutes per case
II.III Main Screens of Performance:

In the following pictures, the main functional screens of the MIRAS performance are shown.

**Screen 1. MIRAS cover**

This is the first screen of MIRAS which shows an animation of contrast 2-D echocardiogram demonstrating the meaning of the software. Meanwhile, it is waiting for a password input.
Screen 2. Basic function menu of MIRAS

This screen gives an basic menu of MIRAS functions with an equipment animation that helps the user to understand the connection of devices and their main performances.

Screen 3. An example system operation screen

This kind of screen appears throughout MIRAS operations. It consists of operation functions, instructions, commands input, system status, and operation process which help users to operate MIRAS.
The following screens are shown when classification results are obtained:

**Screen 4. Basic parameters**

This screen gives the basic parameters values of a case.

**Screen 5. About classification tree**

This screen gives the MR definition, hierarchical classification tree structure, and the number of features at each node.
Screen 6. Classification probabilities and result

This screen gives the prevalence of each class, the probability of each class and each pooled class, and the classification result. It helps the user to understand the classification process of a case.

Screen 7. MR assessment report

This is the last screen of a case assessment. It gives the computer classification result, authorized classification result, the case name. It also gives the probabilities of all the classes in decreasing order, therefore, helping the user to analysis the class ambiguity.

Besides, features values can also be shown when needed.
Appendix III: DT2851, DT2858 and DT-IRIS

INTRODUCTION

DT2851

1.1 DT2851

The DT2851 High Resolution Frame Grabber has eight 256 by 8-bit RAM input look-up tables, two 512 by 512 by 8-bit high speed frame-store memory buffers, eight 256 by 24-bit RAM output look-up tables, and three 8-bit video RGB outputs (one each for red, green, and blue). The DT2851-60Hz is a high resolution frame grabber that digitizes RS-170, RS-330, NTSC, or slow-scan compatible inputs. The DT2851-50Hz digitizes CCIR, PAL, or slow-scan inputs.

The DT2851 frame grabber digitizes an entire video frame in 1/30 (1/25 for the DT2851-50Hz) of a second. The video input is ac-coupled, and then dc-restored. A 3.58MHz (4.43MHz for the DT2851-50Hz) filter is available to remove the color burst and chrominance signals present in the NTSC and PAL signals. This color information would otherwise show up as interference patterns. The video signal is converted at a 10MHz rate into 480 lines (512 lines for DT2851-50Hz) by 512 pixels per line by 8 bits per pixel. The 8-bit information for each pixel represents one of 256 possible gray levels.

The 8-bit input values are sent through one of eight software-selectable input look-up tables stored in on-board RAM. These input look-up tables are used for the LUT processor, for the video input, and for the input from an auxiliary frame processor.

The two 512 by 512 by 8-bit frame-store memory buffers (256Kbytes each) allow the digitized input signal to be stored in memory where it can then be accessed over the IBM Personal Computer AT bus at normal microcomputer bus speeds. These frame-store memory buffers are jumper-selectable to occupy any of the thirty-two 512Kbyte blocks within the 16 Mbyte IBM Personal Computer AT memory space.

The two frame-store memory buffers are memory-mapped into the IBM Personal Computer AT memory space and can be accessed at any time using normal memory instructions. Frame memory operations are controlled by hardware timing circuits and arbitrators. Video input, video output, and normal memory access can all be done simultaneously.
INTRODUCTION

Figure 1-1 shows a block diagram for the DT2851. The DT2851 has two external video ports (one for input and one for output). These ports can be used to transfer video data in and out of the frame-store memory buffers at video speeds using two-way asynchronous handshaking protocols. The DT2851 can access a high-speed video processor (the DT2858 Auxiliary Frame Processor, for example) through these ports.

The DT2851 frame grabber can perform real-time arithmetic and logic image processing operations including AND, OR, XOR, frame averaging, frame addition and subtraction, and multiplication and division. These operations are accomplished on-board through the use of a look-up table (LUT) processor. The LUT processor consists of a feedback path which allows a frame of data from either frame-store memory buffers to be operated upon by one of eight 256 by 8-bit input LUTs, and returned to either of the frame-store buffers and displayed, all in 1/30 (1/25 for the DT2851-50Hz) second.

Eight 256 by 24-bit RAM output look-up tables define eight sets of output attributes (intensity and color) for each of the 256 possible byte values that can be contained in the 8-bit memory. This allows eight independent sets of any 256 display attributes to be chosen out of a total 16 million possible attributes for use in the output display.

The DT2851 makes use of eight 16-bit I/O registers for control and status purposes. The base I/O address is jumper-selectable to one of 16 addresses within the IBM Personal Computer AT’s 10-bit I/O address space.
FIGURE 1-1: DT2851 BLOCK DIAGRAM
FIGURE 3-1: JUMPER LOCATIONS
UNPACKING AND CONFIGURATION

3.3.1 MEMORY BASE ADDRESS CONFIGURATION

The DT2851's frame-store memory can be set to any one of 32 possible memory blocks (512Kbytes) on the IBM Personal Computer AT. The Memory Base Address is configured using five on-board jumpers, W1, W2, W3, W4, and W5 (see Figure 3-1), which control address bits 19 through 23, respectively.

Figure 3-2 indicates the settings of jumpers W1 through W5, and how these jumpers appear in factory configuration. A jumper installed sets a logical zero for the bit it controls. The factory-configured Memory Base Address is A00000 (hex).

NOTE

Ensure that the addressing of other memory devices does not conflict with the factory-configured address. If the memory address will be changed from the factory-shipped configuration, first verify that the address space is available, and that this address does not conflict with that of any other memory device during board operation.

<table>
<thead>
<tr>
<th>A</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>22</td>
<td>21</td>
<td>20</td>
<td>19</td>
<td>18</td>
<td>17</td>
</tr>
</tbody>
</table>

Decoded on board

<table>
<thead>
<tr>
<th>W5</th>
<th>W4</th>
<th>W3</th>
<th>W2</th>
<th>W1</th>
</tr>
</thead>
</table>

FIGURE 3-2: MEMORY BASE ADDRESS SELECTION

3-6
The 32 possible Memory Base Address configurations are given in Table 3-1.

**TABLE 3-1: AVAILABLE MEMORY BASE ADDRESSES**

<table>
<thead>
<tr>
<th>MEMORY ADDRESS (HEX)</th>
<th>W1</th>
<th>W2</th>
<th>W3</th>
<th>W4</th>
<th>W5</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>In</td>
<td>In</td>
<td>In</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>080000</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>100000</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>180000</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>200000</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>280000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>300000</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
<td>In</td>
</tr>
<tr>
<td>380000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>400000</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>480000</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>500000</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>580000</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>600000</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>680000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>700000</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>780000</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>800000</td>
<td>In</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>880000</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>900000</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>980000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>A00000 (FC)</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>A80000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>B00000</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>B80000</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>C00000</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>C80000</td>
<td>Out</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>D00000</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>D80000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>E00000</td>
<td>In</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>E80000</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>F00000</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>F80000</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
<td>Out</td>
</tr>
</tbody>
</table>

1. Memory base addresses 000000 (hex) and 080000 (hex) should never be used as normal system memory addresses. These locations are shown for reference only.
UNPACKING AND CONFIGURATION

CONFIGURATION

3.3.2 I/O REGISTER EASE ADDRESS CONFIGURATION

The I/O Base Address is the lowest I/O address used by the DT2851. Four jumpers select the I/O Base Address: W6, W7, W8, and W9 (see Figure 3-1). These jumpers are marked in silkscreen. Refer to Chapter 5 for more information on the registers that are assigned to the I/O locations.

The jumpers allow you to set the I/O Base Address anywhere in the 210 (hex) to 3F0 (hex) I/O address space in increments of 20 (hex). Figure 3-3 indicates the settings of jumpers W6 through W9, and how these jumpers appear in the factory configuration. A jumper installed sets a logical zero for the bit it controls. The factory-configured I/O Base Address is 230 (hex).

NOTE

Ensure that the addressing of any I/O devices used do not conflict with the addresses occupied by the I/O registers (I/O base address + 2-hex) and the I/O register address will be changed from the factory-shipped configuration, first verify that the address does not conflict with that of any other I/O device used.

<table>
<thead>
<tr>
<th>2</th>
<th>3</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>7</td>
</tr>
</tbody>
</table>

Always 1 Always 1 Always 0

- I/O bits 10 through 19 are not used.
- Installing a jumper clears the bit, removing a jumper sets the bit.
- Address bits 1, 2, and 3 are decoded on the DT2851 frame grabber to select one of eight on-board registers.
- Address bit 0 is always clear to provide word addresses.

FIGURE 3-3: I/O BASE ADDRESS SELECTION

The 16 available I/O Base Address configurations are given in Table 3-2. All addresses are in hex.
UNPACKING AND CONFIGURATION

CONFIGURATION

TABLE 3-2: AVAILABLE I/O BASE ADDRESSES

<table>
<thead>
<tr>
<th>I/O BASE ADDRESS (HEX)</th>
<th>JUMPERS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>W6</td>
</tr>
<tr>
<td>210</td>
<td>In</td>
</tr>
<tr>
<td>230 (FC)</td>
<td>Out</td>
</tr>
<tr>
<td>250</td>
<td>In</td>
</tr>
<tr>
<td>270</td>
<td>Out</td>
</tr>
<tr>
<td>290</td>
<td>In</td>
</tr>
<tr>
<td>2B0</td>
<td>Out</td>
</tr>
<tr>
<td>2D0</td>
<td>In</td>
</tr>
<tr>
<td>2F0</td>
<td>Out</td>
</tr>
<tr>
<td>310</td>
<td>In</td>
</tr>
<tr>
<td>330</td>
<td>Out</td>
</tr>
<tr>
<td>350</td>
<td>In</td>
</tr>
<tr>
<td>370</td>
<td>Out</td>
</tr>
<tr>
<td>390</td>
<td>In</td>
</tr>
<tr>
<td>3B0</td>
<td>Out</td>
</tr>
<tr>
<td>3D0</td>
<td>In</td>
</tr>
<tr>
<td>3F0</td>
<td>Out</td>
</tr>
</tbody>
</table>

1. IBM I/O cards may occupy these addresses. They are shown for reference only.

3.3.3 INTERRUPT PRIORITY

The DT2851 interrupts the processor on either of two conditions: when BUSY (bit 7 of INCSR1) clears and DONEINT (bit 6 of INCSR1) is set; or when VSYNC (bit 15 of OUTCSR) changes from 0 to 1 and SYNClNT (bit 14 of OUTCSR) is set. Chapter 5 contains details of the BUSY, DONEINT, VSYNC, and SYNClNT bits. Table 3-3 details the jumper configuration for selecting the interrupt priority. The order of priority for the four available levels is 10 (highest), 15, 3, and then 5 (lowest). The factory configuration is interrupt level 15.

TABLE 3-3: INTERRUPT PRIORITY SELECTION

<table>
<thead>
<tr>
<th>INTERRUPT LEVEL</th>
<th>JUMPERS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>W10</td>
</tr>
<tr>
<td>15 (FC)</td>
<td>Out</td>
</tr>
<tr>
<td>10</td>
<td>Out</td>
</tr>
<tr>
<td>5</td>
<td>Out</td>
</tr>
<tr>
<td>3</td>
<td>In</td>
</tr>
<tr>
<td>None</td>
<td>Out</td>
</tr>
</tbody>
</table>

3-10
3.3.4 VIDEO INPUT

The following sections describe the jumpers that affect:

- Chrominance Filter Enable
- Bandwidth Limit
- Ac/dc-coupling and Dc Restoration
- 75 ohm Termination
- Input Offset Level and Range

3.3.4.1 Chrominance Filter Enable

Composite color video signals encode the chrominance (color) information in the phase and amplitude of a high frequency, which is added to the grey scale intensity signal. If this composite signal is directly digitized, the chrominance signal will show up in the digitized black and white picture as an interference pattern. Thus, when digitizing a black and white picture from a color source, the chrominance signal must be filtered first.

With color input sources such as NTSC and PAL, jumper W17 should be installed to enable the chrominance filter. The chrominance signal is at 3.48MHz for NTSC signals, and at 4.43MHz for PAL signals. The filter frequency can be adjusted using variable capacitor C59 to obtain the greatest degree of filtering.

With monochrome video sources (RS-170, RS-330, or CCIR), the highest resolution is achieved using the highest possible input bandwidth. For this reason, jumper W17 should be removed to disable the chrominance filter.

Table 3-4 shows the jumper configuration to enable or disable the filter.
UNPACKING AND CONFIGURATION

TABLE 3-4: CHROMINANCE FILTER ENABLE

<table>
<thead>
<tr>
<th>CHROMINANCE FILTER</th>
<th>JUMPER W17</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.58MHz or 4.43MHz</td>
<td></td>
</tr>
<tr>
<td>Enabled (FC)</td>
<td>In</td>
</tr>
<tr>
<td>Disabled</td>
<td>Out</td>
</tr>
</tbody>
</table>

3.3.4.2 Bandwidth Limiting

The bandwidth of the video input amplifier can be adjusted with a jumper and a user-configurable capacitor. The maximum possible bandwidth is 4.5MHz, and this should be used with high quality black and white cameras. When using lower quality input signals from color sources and VCR's, use a lower bandwidth to reduce noise levels since they typically have a bandwidth of only 3MHz.

With high quality black and white cameras, W20 should not be installed. This results in the maximum input bandwidth of 4.5MHz.

When using color or VCR inputs, W20 should be installed and a 68 pf ceramic cap should be installed at C77. This produces an input bandwidth of 3MHz.

TABLE 3-5: BANDWIDTH ADJUSTMENT

<table>
<thead>
<tr>
<th>BANDWIDTH</th>
<th>W20</th>
<th>C77</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5 MHz</td>
<td>Out</td>
<td>X</td>
</tr>
<tr>
<td>3.0 MHz(FC)</td>
<td>In</td>
<td>In</td>
</tr>
</tbody>
</table>

X = Irrelevant

3.3.4.3 AC/DC-coupling And DC Restoration

Standard video inputs require ac-coupling with dc restoration and sync stripping. Slow-scan inputs should be dc-coupled with no dc restoration and no sync stripping.

For RS-170, RS-330, NTSC, CCIR, and PAL inputs, jumpers W15 and W16 should be removed. The input signal is ac-coupled, and the zero level is set at the sync tip of the input signal. The sync signal is extracted and is used for external timing synchronization.
UNPACKING AND CONFIGURATION

When using non-standard slow-scan inputs, jumpers W15 and W16 should be installed. The input signal is dc-coupled, with no dc restoration or sync stripping. The input is buffered and sent directly to the A/D converter. Synchronization must occur using the external timing inputs provided on the J2 connector.

When using a dc-coupled input, the input should NEVER go negative, and the board will operate best if no offset voltage is used.

Table 3-6 shows the configuration for ac/dc-coupling, and dc restoration. In this configuration, slow-scan input timing is controlled by the inputs provided on the J2 connector, and video timing for the output must be selected to use the internal crystal-generated timing.

### TABLE 3-6: AC/DC-COUPLING, DC RESTORATION SELECTION

<table>
<thead>
<tr>
<th>COUPLING AND RESTORATION CONFIGURATION</th>
<th>JUMPERS W15</th>
<th>W16</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC-coupled, DC Restoration, and Sync Stripping (FC)</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>DC-coupled, No DC Restoration, No Sync Stripping</td>
<td>In</td>
<td>In</td>
</tr>
</tbody>
</table>

#### 3.3.4.4 75 Ohm Termination

Standard video inputs generally require a 75 ohm terminating resistance. In special cases, it may be desirable to use a standard video input with an unterminated, high impedance input. Slow-scan inputs are non-standard, and may require termination values other than 75 ohms.

For RS-170, RS-330, NTSC, CCIR, and PAL inputs, jumper W14 should be installed, providing a 75 ohm termination. When using slow-scan devices which require a termination other than 75 ohms, jumper W14 should be removed and the proper termination should be provided externally (see Section 3.3.4.5). Table 3-7 shows the termination configuration. Factory-configuration is with W14 installed.
TABLE 3-7: 75 OHM TERMINATION SELECTION

<table>
<thead>
<tr>
<th>TERMINATION CONFIGURATION</th>
<th>W14</th>
</tr>
</thead>
<tbody>
<tr>
<td>75 ohm Input Termination (FC)</td>
<td>In</td>
</tr>
<tr>
<td>High Impedance Input Termination</td>
<td>Out</td>
</tr>
</tbody>
</table>

3.3.4.5 Input Offset Level And Range

For RS-170, RS-330, NTSC, CCIR, and PAL inputs, jumper W18 should be installed, jumper W19 should be removed, and no resistor should be installed at R32. This results in the digitization of the active video between the black level at 0.340 volts and the saturated white level at 1.000 volts, with voltages referenced to the dc restored sync tip level.

For dc-coupled slow-scan inputs, two fixed ranges are available. In addition, the input amplifier gain can be changed by installing a precision resistor (RN55D). If the particular input used does not match any of these ranges, you must amplify it externally.

When using a dc-coupled input, the input should NEVER go negative, and the board will operate best if no offset voltage is used.

With jumper W18 installed, jumper W19 removed, and no resistor installed at R32, the digitized input range is 0.340 volts to 1.000 volts.

With jumper W18 removed, jumper W19 installed, and no resistor installed at R32, the digitized input range is 0.000 volts to 0.660 volts.

With jumper W18 removed, jumper W19 installed, and a gain resistor installed at R32, the digitized input range is 0.000 volts to V volts, where V can range from 0.660 volts to 1.320 volts. The value (R) of the resistor to be used can be calculated using the following equation:

\[ R = \frac{(249 \times V \text{ range}) - 164.34}{1.32 - V \text{ range}} \]

Where:

1. R is in ohms
UNPACKING AND CONFIGURATION

2. R must be an RN55D precision resistor

3. V is the maximum voltage level of the video input signal (between 0.660 and 1.320 volts).

Table 3-8 shows the input range configuration. Factory-configuration is for a standard ac-coupled signal.

<table>
<thead>
<tr>
<th>INPUT RANGE</th>
<th>W18</th>
<th>W19</th>
<th>R32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard AC-coupled Signal, 0.340V to 1.000V</td>
<td>In</td>
<td>Out</td>
<td>Out</td>
</tr>
<tr>
<td>Slow-scan DC-coupled Signal, 0.340V to 1.000V</td>
<td>In</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>Slow-scan DC-coupled Signal, 0.000V to 0.660V</td>
<td>Out</td>
<td>In</td>
<td>Out</td>
</tr>
<tr>
<td>Slow-scan DC-coupled Signal, 0.000V to 0.560V-1.320V</td>
<td>Out</td>
<td>In</td>
<td>264.5Ω</td>
</tr>
<tr>
<td>Slow-scan DC-coupled Signal, 0.000V to 1.000V</td>
<td>Out</td>
<td>In</td>
<td></td>
</tr>
</tbody>
</table>

1. Use the formula to calculate the resistance needed to digitize the full video input voltage range.

3.4 JUMPER SUMMARY

Table 3-9 lists the jumpers in the factory-configuration. All jumpers are user-configurable, and are explained in this chapter.
## Table 3-9: Jumper Summary

<table>
<thead>
<tr>
<th>Jumper</th>
<th>FC</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>In</td>
</tr>
<tr>
<td>W2</td>
<td>In</td>
</tr>
<tr>
<td>W3</td>
<td>Out</td>
</tr>
<tr>
<td>W4</td>
<td>In</td>
</tr>
<tr>
<td>W5</td>
<td>Out</td>
</tr>
<tr>
<td>W6</td>
<td>Out</td>
</tr>
<tr>
<td>W7</td>
<td>In</td>
</tr>
<tr>
<td>W8</td>
<td>In</td>
</tr>
<tr>
<td>W9</td>
<td>In</td>
</tr>
<tr>
<td>W10</td>
<td>Out</td>
</tr>
<tr>
<td>W11</td>
<td>Out</td>
</tr>
<tr>
<td>W12</td>
<td>Out</td>
</tr>
<tr>
<td>W13</td>
<td>In</td>
</tr>
<tr>
<td>W14</td>
<td>In</td>
</tr>
<tr>
<td>W15</td>
<td>Out</td>
</tr>
<tr>
<td>W16</td>
<td>Out</td>
</tr>
<tr>
<td>W17</td>
<td>In</td>
</tr>
<tr>
<td>W18</td>
<td>In</td>
</tr>
<tr>
<td>W19</td>
<td>Out</td>
</tr>
<tr>
<td>W20</td>
<td>In</td>
</tr>
</tbody>
</table>
ARCHITECTURE AND PROGRAMMING PRINCIPLES

INTRODUCTION

5.1 INTRODUCTION

This chapter describes the architecture and programming principles associated with the DT2851. Detailed information on the DT2851's registers is also provided.

5.2 MEMORY ACCESS

The frame-store memory consists of two consecutive 256Kbyte memory buffers which occupy a user-selected 512Kbyte block of the IBM Personal Computer AT memory address space. The memory can be accessed at any time using standard 8-bit or 16-bit IBM Personal Computer AT memory access operations.

In order for a computer to process an image, it must first obtain a translation of the image in numeric notation. This translation process is called digitizing. The image is subdivided into a rectangular grid of horizontal lines, each made up of adjacent pixels.

The Memory Base Address contains the digitized value for the left-most pixel (0) of the top-most even line (0) of frame-store memory buffer 0. The following 511 (byte) addresses contain the remaining pixels of that line, in order from left to right (0 to 511). Memory Base Address + 512 contains the first pixel (0) on the first odd line (1), and so on. The first pixel of frame-store memory buffer 1 is located at Memory Base Address + 256,144 (256Kbytes). This memory arrangement is the same whether the video acquisition is done interlaced or not. In an interlaced acquisition, the even lines (0, 2, 4, and so on) are digitized in the first pass and the odd lines (1, 3, 5, and so on) are digitized in the subsequent pass. But the frame-store memory buffer still contains the lines in order (0, 1, 2, 3, and so on).

On a DT2851-50Hz, the lines number from 0 to 511, from the top to the bottom of the screen. The pixels which make up the lines number from 0 to 511 from left to right across the screen. This translates to a resolution of 512 lines by 512 pixels, thereby using the entire 512Kbyte memory space. See Figure 5-1.

The DT2851-60Hz has the same memory capacity as the DT2851-50Hz and can also store an image of 512 lines by 512 pixels. But the video format for RS-330, RS-170, and NTSC makes the displayable

5-2
range only 480 lines (0 to 479) by 512 pixels. The lines numbered 480 to 511 display below the bottom of the screen and are not visible. See Figure 5-1.

FIGURE 5-1: SCREEN COORDINATES

Each pixel has an address identified on a rectangular grid represented by the coordinates of the line number and the pixel number in the line. The brightness of the image at that location is sampled by the flash A/D and quantized. This operation permits each pixel to be identified by a numeral which represents the brightness value of the pixel at that location. When all pixels are evaluated, the entire image can be represented by a rectangular array of integers. This image representation can then be processed by the DT2851, or stored in memory for later retrieval and processing.

All 8 bits of the pixel value are grouped into four sets for write protection: bit 0; bit 1; bits 2 and 3; and bits 4, 5, 6, and 7. When protected, data bits in the protected bit plane cannot be written to from the IBM Personal Computer AT bus, nor are they changed by video acquisition operations.

The frame-store memory has two external video ports (one for input and one for output), which can be used to transfer video data in (through the input LUTs) and out of the frame-store memory buffers asynchronously. The transfer of data over this bus is done a complete frame at a time, to or from frame-store memory buffer 0 or frame-store memory buffer 1. These ports may be used to interface the DT2851 with a high-speed video frame processor (such as the DT2858 Auxiliary Frame Processor).  

5-3
ARCHITECTURE AND PROGRAMMING PRINCIPLES
MEMORY ACCESS

An LUT processor exists such that a frame of data can be read out of either frame-store memory buffer, transformed through the input look-up table, and written back into either frame-store memory buffer in a single-frame time. This allows real-time 8-bit arithmetic and logic image processing to be done at full video speeds (for example, two 4-bit images can be subtracted giving an 8-bit result). See Section 5.4 (Operating Modes) for more information.

5.3 HARDWARE REGISTERS

Control of the DT2851 occurs through eight consecutive 16-bit wide read/write I/O registers. These registers can only be accessed using 16-bit I/O operations available on the IBM Personal Computer AT. The board does not respond to 8-bit bus operations.

All registers are fully described in this chapter, and are listed in Table 5-1. Register addresses are described as offsets of the user-configurable I/O Base Address. See Chapter 3 for information on selecting an I/O Base Address. Some of the registers can be accessed only under certain conditions. See the individual register discussions for more information.

NOTE

There is no default or reset state for the register bits. All bits should be initialized by software after power-up or a system reset.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
HARDWARE REGISTERS

TABLE 5-1: REGISTER ADDRESS ALLOCATIONS

<table>
<thead>
<tr>
<th>BASE ADDRESS + (HEX)</th>
<th>REGISTER NAME</th>
<th>REGISTER FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Video Input Control/Status Register 1 (INCSR1)</td>
<td>Controls the video input</td>
</tr>
<tr>
<td>2</td>
<td>Video Input Control/Status Register 2 (INCSR2)</td>
<td>Controls the video input</td>
</tr>
<tr>
<td>4</td>
<td>Video Output Control/Status Register (OUTCSR)</td>
<td>Controls the video output</td>
</tr>
<tr>
<td>6</td>
<td>Cursor (CURSOR)</td>
<td>Contains cursor pixel and line.</td>
</tr>
<tr>
<td>8</td>
<td>Index (INDEX)</td>
<td>Contains the LUT index</td>
</tr>
<tr>
<td>A</td>
<td>Input Look-Up Table Entry (INLUT)</td>
<td>Contains the LUT entry</td>
</tr>
<tr>
<td>C</td>
<td>RedGreen Output Look-Up Table Register (REDGRN)</td>
<td>Contains the red &amp; green LUT entry</td>
</tr>
<tr>
<td>E</td>
<td>Blue Output Look-Up Table (BLUE)</td>
<td>Contains the blue LUT entry</td>
</tr>
</tbody>
</table>

The following sections contain explicit descriptions of each interface register and its program applications.
ARCHITECTURE AND PROGRAMMING PRINCIPLES

5.3.1 VIDEO INPUT CONTROL/STATUS REGISTER 1 (INCSR1), BASE+0

The INCSR1 is a read/write register that interfaces to and controls the DT2851 frame grabber. This register can be read and written at any time. Figure 5-2 gives the bit assignments of the INCSR1.

![Figure 5-2: INCSR1 BIT ASSIGNMENTS](image)

**FIGURE 5-2: INCSR1 BIT ASSIGNMENTS**

Bits 15-8 - RESERVED, Read as 1/Write ignored
These bits are unassigned and read back as ones. All writes to these bits are ignored.

Bit 7 - BUSY, R/W
Setting BUSY starts the operation selected by MODE (bits 4 through 6 in the INCSR2). While busy remains set, the selected operation is in progress. If the ENSTOP bit is set, BUSY clears automatically at the completion of an operation, otherwise BUSY remains set and multiple operations occur. You may bring the board to an immediate stop by first writing a '1' to the ENSTOP bit, then writing a '0' to the BUSY bit. This is not recommended except when initializing the board. BUSY is clear on power-up. BUSY cannot be set when in a "reserved" operating mode (MODE equals 010 or 011 binary).

The VIDEO IN and FEEDBACK operations are synchronized with the board's video timing. When BUSY is set, VIDEO IN and FEEDBACK modes do not actually begin until the end of an even field vertical sync occurs. BUSY is automatically cleared at the beginning of the
ARCHITECTURE AND PROGRAMMING PRINCIPLES
INCSRI

next even field vertical sync. If ENSTOP is clear, BUSY sets again to perform another operation. ENSTOP must be set for BUSY to remain clear and to stop operations. When ENSTOP is clear, a done interrupt is generated each time BUSY temporarily clears (if interrupts are enabled).

The PORT IN, PORT OUT, and SLOW-SCAN modes all begin operations on pixel 0, line 0, after BUSY is set and the appropriate external control signals have occurred. Pixel sequencing is non-interlaced: line 1 is operated on immediately after line 0. After 256,144 (256K) pixel operations have occurred, BUSY is automatically cleared. If ENSTOP is clear, BUSY sets again to perform another operation. ENSTOP must be set for BUSY to remain clear and to stop operations. When ENSTOP is clear, a done interrupt is generated each time BUSY temporarily clears (if interrupts are enabled).

If ENSTOP is clear, BUSY remains set You cannot clear BUSY unless you set ENSTOP first. To do this, write to INCSR1 to set ENSTOP (BUSY clears and stays after the operation is complete). If no operation is pending, do another write to INCSR1 to clear BUSY (keep ENSTOP set).

Section 5.4 (Operating Modes) has more details.

Bit 6 - INTERRUPT ON DONE (DONEINT), R/W
When set, DONEINT enables an interrupt to occur at the end of any operation. This interrupt occurs at the same time that BUSY is automatically cleared. When an interrupt occurs, this bit and SYNCINT (bit 14 of OUTCSR) must be cleared to clear the interrupt, which is latched on to the board.

Bit 5 - EXTERNAL TRIGGER (EXTTRG), R/Write ignored
This bit indicates the state of the external digital input signal on J2 pin 3. It is also used to control the start of the SLOW SCAN operation. This is not an enable bit, but rather reflects the logical state of the external trigger input.

Bit 4 - FRAME PROCESSOR* (FRPROC*), R/Write ignored
When 0, this bit indicates the presence of the DR2858 auxiliary frame processor board. When 1, the frame processor is not connected.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
INCSR1

Bit 3 - ENABLE STOP (ENSTOP), R/W
When set, writing a 1 to BUSY (bit 7) enables a single operation. The operation starts as specified in the description of BUSY. When the operation is completed, BUSY automatically clears. When ENSTOP is clear, BUSY automatically sets, regardless of the status of board operations, and BUSY remains set despite any attempts to clear BUSY by writing to the INCSR1 register. This allows multiple operations without having to set BUSY each time.

Section 5.4 (Operating Modes) has more information.

Bits 2-0 - INPUT LUT SELECT (ISEL2, ISEL1, ISEL0), R/W
These bits select one of eight input look-up tables to be used during input operations. All input data is transformed through these tables before being written into a frame-store memory buffer.

<table>
<thead>
<tr>
<th>INPUT LUT</th>
<th>INPUT ISEL2</th>
<th>LUT SELECT BITS</th>
<th>ISEL1</th>
<th>ISEL0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

5-8
5.3.2 VIDEO INPUT CONTROL/STATUS REGISTER 2 (INCSR2), BASE+2

The INCSR2 register is a read/write register which interfaces to and controls the DT2851. This register can only be written to if BUSY (bit 7 of INCSR1) is clear; otherwise writes are ignored. INCSR2 can be read at any time.

![Buffer Select WP3 WP1 Diagram]

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Read as 1

Mode WP2 WP0

**FIGURE 5-3: INCSR2 BIT ASSIGNMENTS**

**Bits 15-8** - RESERVED, Read as 1/Write ignored. These bits are unassigned and read back as ones. All writes to these bits are ignored.

**Bit 7** - BUFFER SELECT (BUFSEL), R/W
This bit determines which frame-store memory buffer is used during execution of any of the operating modes. When set, this bit specifies frame-store memory buffer 1. When clear, this bit specifies frame-store memory buffer 0. (Note: a bit in the OUTCSR, DSBUF, selects the buffer used for display and feedback data.)

**Bits 6-4** - MODE (MODE2, MODE1, MODE0), R/W
These bits control which operation occurs when BUSY (bit 7 of INCSR1) is set. These operating modes are explained in Section 5.4.

**Bit 3** - WRITE PROTECT 3 (WP3), R/W
When set, this bit disables all write operations to bit planes 4, 5, 6, and 7 of the frame-store memory buffers. Bus reads and writes are affected, as well as video acquisition. For example, if WP3 is set (and WP0, WP1, and WP2 are clear), you can change bits 0, 1, 2, and 3; but not bits 4, 5, 6, or 7 of the frame-store memory buffers.

**Bit 2** - WRITE PROTECT 2 (WP2), R/W
When set, this bit disables all write operations to bit planes 2 and 3 of the frame-store memory buffers. Bus reads and writes are affected, as well as video acquisition. If WP2 is set you cannot change the states of bits 2 and 3 in the frame-store memory buffers.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
INCSR2

Bit 1 - WRITE PROTECT 1 (WP1), R/W
When set, this bit disables all write operations to bit plane 1 of the frame-store memory buffers. Bus reads and writes are affected, as well as video acquisition. If WP1 is set you cannot change the state of bit 1 in the frame-store memory buffers.

Bit 0 - WRITE PROTECT 0 (WP0), R/W
When set, this bit disables all write operations to bit plane 0 (LSB) of the frame-store memory buffers. Bus reads and writes are affected, as well as video acquisition. If WP0 is set you cannot change the state of bit 0 in the frame-store memory buffers.

<table>
<thead>
<tr>
<th>MODES</th>
<th>MODE2 BIT 6</th>
<th>MODE1 BIT 5</th>
<th>MODE0 BIT 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feedback</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Video In</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Load LUT</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Slow-scan</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Port Out</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Port In</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

IMPORTANT.
The write protect bits (WP3, WP2, WP1, WP0) are not cleared on reset or power-up. It may be impossible to write to the DT2851 frame grabber’s frame-store memory buffers until after the INCSR2 is initialized.

5-11
5.3.7 REDGREEN OUTPUT LOOK-UP TABLE REGISTER, BASE+C

The REDGRN register is a read/write register which contains the data relating to the red and green outputs of the selected look-up table entry.

Bits 0 through 7 contain the data of the RED output look-up table selected by the values of OSEL2, OSEL1, and OSEL0 (specified by INDEX). Bits 8 through 15 contain the data of the GREEN output look-up table selected by the values of OSEL2, OSEL1, and OSEL0 (specified by INDEX). If the bits are all zeros, the lowest intensity (black) is produced; if they are all ones full intensity is produced. This register can be accessed only when the board is in the Load LUT mode (MODE = 100).

<table>
<thead>
<tr>
<th>MSB</th>
<th>LSB</th>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Green LUT Data Red LUT Data

FIGURE 5-8: REDGRN BIT ASSIGNMENTS

5.3.8 BLUE OUTPUT LOOK-UP TABLE REGISTER, BASE+E

The BLUE register is a read/write register which contains the data relating to the blue output of the selected LUT (specified by OSEL2, OSEL1, and OSEL0 of OUTCSR) entry (specified by INDEX). If the bits are all zeros, the lowest intensity (black) is produced; if they are all ones full intensity is produced. This register can be accessed only when the board is in the Load LUT mode (MODE = 100).

<table>
<thead>
<tr>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>14</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Read as 1 Blue LUT Data

FIGURE 5-9: BLUE BIT ASSIGNMENTS
ARCHITECTURE AND PROGRAMMING PRINCIPLES

5.3.3 VIDEO OUTPUT CONTROL/STATUS REGISTER (OUTCSR), BASE+4

The OUTCSR register is a read/write register which interfaces to and controls the video output portion of the DT2851. This register can be read or written at any time.

![Diagram of OUTCSR Bit Assignments]

**FIGURE 5-4: OUTCSR BIT ASSIGNMENTS**

**Bit 15 - VERTICAL SYNC (VSYNC), R/Write ignored**
When set, this bit indicates that vertical sync is occurring in the video timing. Changes to the input and output look-up tables, cursor, and BUSY (bit 7 of INCSR1) may be made without showing up on the output display, if done during vertical sync. VSYNC is set for 10 video lines at the beginning of each field. After this bit changes from 1 to 0, there is an additional 10 video lines before active display begins. This guarantees at least 630 microseconds of blanking after the bit is checked and found to be set.

**Bit 14 - INTERRUPT ON SYNC (SYNCINT), R/W**
When set, this bit allows an interrupt to occur on the next clear to set transition of VSYNC (bit 15 of OUTCSR). When an interrupt occurs, this bit and DONEINT (bit 6 of INCSR1) must be cleared to clear the interrupt, which is latched on the board.

**Bit 13 - FIELD, R/Write ignored**
When this bit is set, it indicates the odd field is being displayed or digitized. When clear, this bit indicates the even field is being displayed or digitized.

5-12

A 83
ARCHITECTURE AND PROGRAMMING PRINCIPLES

Bit 12 - EXTERNAL TRIGGER (EXTTRG), R/Write ignored
This bit indicates the state of the external
digital input signal on J2 pin 3. This is not
an enable bit, but rather reflects the logical
state of the external trigger input. This is a
duplicate of EXTRG in INCSR.

Bits 11-8 - RESERVED, Read as 1/Write ignored
These bits are unassigned and read back as
ones. All writes to these bits are ignored.

Bit 7 - DISPLAY (DISP), R/W
When set, data is taken from the selected
frame-store memory buffer (DISBUF of OUTCSR),
transformed by the selected output look-up
table (OSEL0, OSEL1, and OSEL2 of OUTCSR), and
presented to the inputs of the red, green, and
blue DACs. When clear, a black image is
displayed on the monitor.

Bit 6 - CURSOR (CURS), R/W
When set, a full-screen, cross-hair cursor is
presented to the video outputs to be displayed.
The location of the intersection of the lines
is specified in the CURSOR register. The
cursor is two lines high by two pixels wide.
When clear, no cursor is displayed. DISPLAY
(bit 7 of OUTCSR) must be set in order to
display the cursor. The cursor is displayed
10% whiter than white.

Bit 5 - EXTERNAL TIMING (EXTTIM), R/W
When set, video timing is synchronized to the
input video signal through the phase-locked
loop. When clear, video timing is generated by
an internal crystal-controlled clock.

Bit 4 - DISPLAY BUFFER SELECT (DISBUF), R/W
This bit determines which of the two
frame-store memory buffers is selected for
display and for the feedback data source. When
set, this bit specifies frame-store memory 1.
When clear, this bit specifies frame-store
memory 0.

Bits 3-0 - OUTPUT LUT SELECT
(OSEL3, OSEL2, OSEL1, OSEL0), R/W
These bits select one of eight output look-up
tables to be used for the output display.
OSEL3 is unassigned and reads back as a zero.
All writes to this bit are ignored.

<table>
<thead>
<tr>
<th>TABLE 5-4: OUTPUT LUT SELECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT LUT</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

A . 84
ARCHITECTURE AND PROGRAMMING PRINCIPLES

5.3.4 CURSOR, BASE+6

The CURSOR register contains the pixel and line position of the cursor divided by two. This allows 8 bits for the pixel address (bits 0 through 7) and 8 bits for the line address (bits 8 through 15). The cursor can only be set to even pixels and lines. The resulting full-screen, cross-hair cursor is two pixels wide and two pixels high. This cursor covers not only the selected line and pixel, but the next odd line and pixel as well.

Figure 5-5 gives the bit assignments of the CURSOR register.

```
<table>
<thead>
<tr>
<th>MSB</th>
<th>LSB</th>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Line Address + 2       Pixel Address + 2

FIGURE 5-5: CURSOR BIT ASSIGNMENTS

Examples

If you load CURSOR with 55AA (hex), the lines are 170 (decimal) and 171 (decimal), the pixels are 340 (decimal) and 341 (decimal).

If you want the cursor at lines 120 (decimal) and 121 (decimal) and pixels 256 (decimal) and 257 (decimal), you load CURSOR with:

LINE = 120 * 2 = 60 (decimal), 3C (hex)

PIXEL = 256 * 2 = 128 (decimal), 80 (hex)

CURSOR = 3C80 (hex)
ARCHITECTURE AND PROGRAMMING PRINCIPLES
INDEX

5.3.5 INDEX, BASE+8

The INDEX register is a read/write register which determines the LUT index to be used when reading or writing look-up table entries of the IINLUT, REDGRW, or BLUE registers. This register can be read and written only when the board is in the Load LUT mode (MODE = 100).

![Index Register Diagram]

Read as 1
Select LUT Entry

FIGURE 5-6: INDEX BIT ASSIGNMENTS

5.3.6 INLUT LOOK-UP TABLE ENTRY, BASE+A

The INLUT register is a read/write register which contains the data to be written to or the data that was read from the selected LUT (specified by ISEL2, ISEL1, and ISEL0 of INCSR1) entry (specified by INDEX). This register can be accessed only when the board is in the Load LUT mode (MODE = 100).

![INLUT Register Diagram]

Read as 1
Contain LUT Data

FIGURE 5-7: INLUT BIT ASSIGNMENTS
ARCHITECTURE AND PROGRAMMING PRINCIPLES
OPERATING MODES

5.4 OPERATING MODES

This section describes in detail the operating modes available for the DT2851. These modes are selected with MODE0, MODE1, and MODE2 (bits 4, 5, and 6 of the INCSR2).

<table>
<thead>
<tr>
<th>TABLE 5-5: MODE SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODES</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Feedback</td>
</tr>
<tr>
<td>Video In</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>Load LUT</td>
</tr>
<tr>
<td>Slow-Scan</td>
</tr>
<tr>
<td>Port Out</td>
</tr>
<tr>
<td>Port In</td>
</tr>
</tbody>
</table>

5.4.1 FEEDBACK

The feedback operation allows a frame of data from either frame-store memory buffer to be operated upon by a specified input look-up table and returned to a specified frame-store memory buffer.

When the MODE bits are set to 000, the FEEDBACK operations are selected. When BUSY is set, a complete frame of data is read from the frame-store memory selected by DISBUF (bit 4 of OUTCSR) and sent through the input look-up table selected by the ISEL bits (bits 0 to 3 of INCSR1). It is then written to the frame-store memory buffer selected by BUFSEL (bit 7 of INCSR2).

The Feedback operations are synchronized to the DT2851’s video timing. When BUSY is set, the Feedback operations do not actually begin until the even field vertical sync occurs. If ENSTOP is set, BUSY is automatically cleared at the end of the following odd field vertical sync and the Feedback operation is over.

The Feedback operations occur at full video speed and are interlaced. Line 2 is operated on immediately after line 0 and line 1 is not digitized until the odd video field. On a DT2851-60Hz, only 480 lines are affected.

5-20
ARCHITECTURE AND PROGRAMMING PRINCIPLES
OPERATING MODES

5.4.2 VIDEO IN

When the mode bits are 001 (binary), an 'input signal' is digitized, and data from the video input is fed through the specified input look-up table to the frame-store memory buffer.

When BUSY (bit 7 of INCSR1) is set, a complete frame of data is digitized by the input A/D, sent through the input look-up table selected by the ISEL bits 'bits 0 to 3 in INCSR1', and written to the frame-store memory buffer selected by BUFSEL (bit 4 INCSR2).

The Video In operation is synchronized with the DT2851's video timing. When BUSY is set, the Video In operation does not actually begin until the even field vertical sync occurs. If the ENSTOP bit is set, BUSY is then automatically cleared at the beginning of the next even field vertical sync and the operation is complete.

The Video In operation occurs at full video speed and is interlaced. Line 2 is digitized immediately after line 0 and line 1 is not digitized until the odd video field. On the DT2851-60HZ, only 480 lines are digitized.

5.4.3 LOAD LUT

When the mode bits are 100 (binary), the INDEX, INLUT, REDGRN, and BLUE registers can be accessed to read and write the look-up tables. If the display is on and in the active part of the display, the output attributes in REDGRN and BLUE pointed to by the INDEX register are displayed.

5.4.4 SLOW-SCAN INPUT

Slow-scan mode synchronizes the DT2851 to inputs from slow-scan devices. When the mode bits are 101 (binary), the DT2851 is in slow-scan mode: data from the video input is digitized by the input A/D, fed through the input look-up tables to the frame-store memory buffer, and the slow-scan timing controls are enabled. The timing of the slow-scan operation is controlled by three external timing signals which must be provided on connector J2. These signals are External Trigger*, Clock Enable*, and Scan Clock*. These signals are used to accommodate different frame, line, and pixel digitization rates. External Trigger* is used to
ARCHITECTURE AND PROGRAMMING PRINCIPLES
OPERATING MODES

initiate the digitization of a complete frame. Clock Enable* is used to differentiate between active video and blanking between lines. Scan Clock* is used to establish the rate for digitizing pixels. Slow scan digitization rates may vary from dc to three million pixels per second.

For slow-scan mode, clear EXTTIM (bit 5 in OUTCSR) to select the internal, crystal-controlled clock. This provides a stable video output.

When BUSY (bit 7 of INCSR1) is set, a complete frame of data is digitized by the A/D when a low is detected on External Trigger*: The data is then sent through the input look-up table specified by the ISEL bits (bits 0 to 3 of INCSR1), and written to the frame-store memory buffer selected by BUFSEL (bit 7 of INCSR2). A pixel is digitized on each Scan Clock* high-to-low transition (pin 5 of J2) that occurs while Clock Enable* (pin 1 of J2) is low.

The slow-scan operation begins on pixel 0, line 0 after BUSY is set and the appropriate external control signals have occurred. Pixel sequencing is non-interlaced; line 1 is digitized immediately after line 0. After 262,144 (256K) pixel operations have occurred, BUSY clears itself automatically, if ENSTOP is set. All 512 lines are digitized by this operation, regardless of whether the board is a DT2851-60Hz or DT2851-50Hz. Once digitized, the data is stored on-board and is available for processing and display.

5.4.5 PORT OUT

When the mode bits are 110 (binary), data is sent out over the external output port.

When BUSY (bit 7 of INCSR1) is set, a complete frame of data is read out of the frame-store memory buffer selected by BUFSEL (bit 7 of INCSR2) and is transferred to the external port (J4).

Each pixel transfer is accomplished over the external port using a standard asynchronous handshaking protocol. When the DT2851 drives a valid byte of pixel data to the external port, Request* (pin 13 of J4) goes low. The Request* signal stays low and the data is valid until the external device (such as the DT2858) drives Reply* (pin 15 of J4) low, indicating that the data has been read.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
OPERATING MODES

The Port Out operation begins on pixel 0, line 0 after BUSY is set and the control signals have occurred. Pixel sequencing is non-interlaced. Line 1 is digitized immediately after line 0. After 262,144 (256K) pixel operations have occurred, BUSY clears automatically, if ENSTOP is set. All 512 lines are transferred by this operation, regardless of whether the board is a DT2851-60Hz or DT2851-50Hz.

5.4.6 PORT IN

When the mode bits are 111 (binary), data is acquired from the external input port (J3) and fed through the designated input look-up table to the designated frame-store memory buffer.

When BUSY (bit 7 of INCSR1) is set, a complete frame of data is acquired from the external port, sent through the input look-up table selected by the ISEL bits (bits 0 to 3 of INCSR1), and written to the frame-store memory buffer selected by BUFSEL (bit 7 of INCSR2).

Each pixel transfer is accomplished over the external port using a standard asynchronous handshaking protocol. When the DT2851 is ready to receive a byte of pixel data, Request* (pin 13 of J3) goes low. This signal stays low until the external device (such as the DT2858) drives Reply* (pin 15 of J3) low indicating that valid data is available at the port. The external device should guarantee that the data is valid for as long as Reply* is low. Reply* should remain low until Request* returns high.

The Port In operation begins on pixel 0, line 0 after BUSY is set and the control signals have occurred. Pixel sequencing is non-interlaced. Line 1 is digitized immediately after line 0. After 262,144 (256K) pixel operations have occurred, BUSY clears itself automatically, if ENSTOP is set. All 512 lines are transferred by this operation, regardless of whether the board is a DT2851-60Hz or DT2851-50Hz.

5-23
ARCHITECTURE AND PROGRAMMING PRINCIPLES
MULTIPLE OPERATIONS USING ENSTOP

5.5 MULTIPLE OPERATIONS USING ENSTOP

To perform a single operation, write to INCSR1, setting BUSY (bit 7) and ENSTOP (bit 3). To start multiple operations, write to INCSR1, clearing ENSTOP (bit 3). This automatically sets BUSY when appropriate (see the following discussions).

When ENSTOP (bit 3 of INCSR1) is clear, BUSY (bit 7 of INCSR1) automatically sets again after it temporarily clears at the end of an operation. If DONEINT (bit 6 of INCSR1) is set, the DT285 L interrupts each time BUSY (bit 7 of INCSR1) clears. When ENSTOP (bit 3 of INCSR1) is clear, BUSY (bit 7 of INCSR1) is always set (except for the temporary clear to interrupt the processor). When BUSY is set, you cannot access INC_SR2 or INLUT. This means that during multiple operations, you cannot change the frame-store memory buffer, the operating mode, or the write-protected bits.

To stop multiple operations, write to INCSR1 and set both ENSTOP (bit 3) and BUSY (bit 7). BUSY automatically clears and stays clear at the end of the present operation.

5.6 PROGRAMMING

The following sections describe the typical programming sequences for:

- Board initialization
- Output LUT programming
- Display programming
- Input LUT programming
- Feedback
- Video input
- Slow-scan input
- External port input
- External port output
ARCHITECTURE AND PROGRAMMING PRINCIPLES
BOARD INITIALIZATION

5.6.1 BOARD INITIALIZATION

1. Initialize OUTCSR, display off.
2. Write INCSR1 to set ENSTOP (bit 3).
3. Write INCSR1 to clear BUSY (bit 7). This stops the board. These two steps must be done separately.
4. Program input look-up tables (see Section 5.6.2).
5. Program output look-up tables (see Section 5.6.3).
6. Initialize INCSR2, anything except Load LUT mode.
7. Initialize memory.
8. Write to OUTCSR to select display on, cursor, etc.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
INPUT LUT PROGRAMMING

5.6.2 INPUT LUT PROGRAMMING

If the current display is not to be disturbed, this sequence must be done during vertical sync.

1. Check BUSY (bit 7 of INCSR1). BUSY must be clear to proceed. This should not be done by directly writing to INCSR1, since a set BUSY bit indicates the board is performing an operation. If the board is operating with ENSTOP clear, read INCSR1, write a 1 into ENSTOP (bit 3 of INCSR1), and write the result back to INCSR1. This allows the board to stop normally at the end of operation. Otherwise, the program should either loop until BUSY is clear, or jump to an error routine because there should not be an operation in progress.

2. Read and save the values of INCSR1 and INCSR2.

3. Write to INCSR2, setting the board to Load LUT mode (MODE = 100).

4. Write INCSR1 to set the ISEL bits (bits 0 to 3) to select the input look-up table to be programmed. Do not set BUSY!

5. Write the appropriate value to INDEX, selecting the look-up table entry.

6. Write INLUT with the appropriate input look-up table value in the low byte. The high byte does not matter.

7. Repeat steps 4 through 6 for all input LUT entries to be programmed.

8. Restore INCSR1 and INCSR2 to their original values.
5.6.3 OUTPUT LUT PROGRAMMING

If the current display is not to be disturbed, this sequence must be done during vertical sync.

1. The BUSY bit MUST be clear to proceed. (See Section 5.6.2 for further information.)

2. Save values of INCSR2 and OUTCSR.

3. Write to INCSR2, setting the mode bits to Load LUT (MODE = 100).

4. Write the appropriate value to the OSEL bits in OUTCSR, selecting the look-up table to be loaded.

5. Write the appropriate value to INDEX, selecting the look-up table entry.

6. Write the appropriate output LUT values for red and green to REDGRN, with the value of red in the low byte and the value of green in the high byte.

7. Write BLUE with the value of blue in the low byte. The high byte does not matter.

8. Repeat steps 4 through 7 for all LUT entries to be programmed.

9. Restore original values to INCSR2 and OUTCSR.

5.6.4 DISPLAY PROGRAMMING

1. Program the output look-up table that will be used (see Section 5.6.3).

2. If the cursor is to be displayed, write CURSOR with the desired cursor position.

3. Load the selected frame-store memory buffer with the data to be displayed.

4. If the MODE bits in INCSR2 are set to Load LUT, the single output LUT attribute selected by INDEX is displayed in place of the selected memory data.

5. Write OUTCSR, selecting whether the output display is on, whether the cursor is shown, which memory buffer is to be displayed, and which output look-up table is to be used.
5.6.5 FEEDBACK

1. Program the input look-up table that is to be used (see Section 5.6.2).
2. Check BUSY (bit 7 of INCSR1). BUSY must be clear to proceed. (See Section 5.6.2 for further information.)
3. Write to OUTCSR, setting DISBUF (bit 4) to select the frame-store memory buffer.
4. Write INCSR2 to set MODE (bits 4 to 6) for feedback operation (000 binary), to select the frame-store memory buffer (BUFSEL, bit 7), and to set WP0, WP1, WP2, and WP3 for the desired level of write protection.
5. Write INCSR1 to select the input look-up table (bits 0 to 3) and to set BUSY (bit 7) to start the operation. ENSTOP (bit 3) should be set.
6. Read INCSR1. When BUSY clears, the operation is done.

5.6.6 VIDEO INPUT

1. Program the input look-up table that is to be used (see Section 5.6.2).
2. Check BUSY (bit 7 of INCSR1). BUSY must be clear to proceed. (See Section 5.6.2 for further information.)
3. Write INCSR2 to set MODE bits (bits 4 to 6) for Video In operation (001 binary), to select which frame-store memory buffer (BUFSEL, bit 7) to use, and to set WP0, WP1, WP2, and WP3 to the desired level of write protection.
4. Write INCSR1 to set the ISEL bits (bits 0 to 3) to select the input look-up table and to set BUSY (bit 7) to start the operation. ENSTOP (bit 3) should be set.
5. READ INCSR1. When BUSY clears, the operation is done.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
SLOW-SCAN INPUT

5.6.7 SLOW-SCAN INPUT

1. Program the input look-up table that is to be used (see Section 5.6.2).

2. Check BUSY (bit 7 of INCSR1). BUSY must be clear to proceed. (See Section 5.6.2 for more information.)

3. Write INCSR2 to set MODE bits (bits 4 to 6) for Slow-scan operation (101 binary), to select which frame-store memory buffer (BUFSEL, bit 7) to use, and to set WP0, WP1, WP2, and WP3 to the desired level of write protection.

4. Write INCSR1 to set ISEL bits (bits 0 to 3) to select the input look-up table and to set BUSY (bit 7) to start the operation. ENSTOP (bit 3) should be set.

5. Read INCSR1. When BUSY clears, the operation is done.

5.6.8 EXTERNAL PORT OUTPUT

1. Check BUSY (bit 7 of INCSR1). BUSY must be clear to proceed. (See Section 5.6.2 for further information.)

2. Write INCSR2 to set MODE (bits 4 to 6) to select Port Out operation (110 binary) and to select the frame-store memory buffer (BUFSEL, bit 7) to use. The write protect bits are not used.

3. Write INCSR1 to set BUSY (bit 7) to start the operation. ENSTOP (bit 3) should be set. The input look-up table is not used.

4. Read INCSR1. When BUSY clears, the operation is done.
ARCHITECTURE AND PROGRAMMING PRINCIPLES
EXTERNAL PORT INPUT

5.6.9 EXTERNAL PORT INPUT

1. Program the input look-up table that is to be used (see Section 5.6.2).

2. Check BUSY (bit 7 of INCSR1). BUSY must be clear to proceed. (See Section 5.6.2 for further information.)

3. Write INCSR2 to set the MODE bits (bits 4 to 6) to select Port In operation (111 binary), to select which frame-store memory buffer to use (BUFSEL, bit 7), and to set WPO, WP1, WP2, and WP3 to the desired level of write protection.

4. Write INCSR1 to set ISEL bits (bits 0 to 3) to select the input look-up table and to set BUSY to start the operation. ENSTOP (bit 3) should be set.

5. Read INCSR1. When BUSY clears, the operation is done.
CHAPTER 6
PRINCIPLES OF OPERATION

6.1 INTRODUCTION

The DT2851 frame grabber digitizes an analog video input, stores the image in one of two on-board frame-store memory buffers, and converts the digitized image back into an analog output. This serves two basic functions. The first is common to any frame grabber: to make the frame available as data so a variety of digital image processing techniques can be used to enhance the image, to extract information about the image from it, to alter it; to store the image in system memory; to plot or print the image; or to archive it on disk. The second is to permit the processed image to be displayed on an analog-input monochrome or RGB monitor.

6.2 VIDEO INPUT

The DT2851 operates in real-time. That is, if the DT2851-60Hz is connected to an input conforming to a 60Hz video standard (such as RS-170, RS-330, or NTSC), it digitizes, stores, and displays images at a rate of 30 frames per second (the DT2851-50Hz provides real-time operation at the rate of 25 frames per second with CCIR or PAL inputs).

The video input circuitry is compatible with an unusually wide range of video standards. The DT2851-60Hz accepts RS-170, RS-330, or NTSC inputs. The DT2851-50Hz accepts CCIR or PAL inputs. Input circuits are ac-coupled, as is required for standard video inputs, and dc restored to set the zero level accurately at the A/D converter.

Analog-to-digital conversion is provided by an 8-bit flash converter, operating at a frequency of approximately 10MHz. From the A/D converter, the pixel data passes through one of eight input look-up
PRINCIPLES OF OPERATION
VIDEO INPUT

tables and then to one of two 256Kbyte frame-store memory buffers.

6.3 PHASE-LOCKED LOOP

For standard video input signals, timing is provided by a phase-locked loop circuit operating either from a crystal controlled 15.75kHz time base (15.625kHz for the DT2851-50Hz), or from a sync signal stripped from the composite video input. Most phase-locked loop circuits work perfectly well with the relatively clean signals presented by video cameras, television broadcasts, or slow-scan devices, but fail down completely when acquiring images from a VCR. The phase-locked loop circuit on the DT2851 frame grabber was designed specifically to provide stable sampling of VCR images, and therefore is relatively immune to occasional missing or noisy sync pulses.

6.4 SLOW-SCAN INPUTS

In addition, the DT2851 frame grabber has control inputs available on J2 (External Trigger*, Scan Clock*, Clock Enable*) to accept non-standard video inputs, such as input from slow-scan devices. External Trigger* indicates the beginning of a frame, and is used to start frame acquisition as would the end of an RS-170 vertical sync interval. Clock Enable* specifies the active line period. Scan Clock* indicates when to digitize each pixel, and triggers the A/D converter. Slow-scan digitization rates may vary from dc to 3MHz. While slow-scan signals from different sources (a scanning electron microscope for example, or computed tomography scanners) may provide different control signals, the control inputs on the DT2851 frame grabber can be used directly or they can be modified easily to accommodate most such input devices. Chapter 2 lists the specifications for the slow-scan inputs.

For non-standard inputs, jumper-selectable input ranges are provided (see Chapter 3) which accommodate different reference black and white voltage levels to optimize gray-scale resolution. These inputs are dc-coupled, and need no dc restoration.
6.5 NOTCH FILTER

Image acquisition on the DT2851 frame grabber is inherently monochrome. When composite color signals are presented (as from a VCR or a composite color camera), a jumper-selectable notch filter (see Chapter 3) removes the color burst and chrominance signals which otherwise would create interference patterns on the acquired image. When a monochrome source is used, the filter can be disabled and the DT2851 frame grabber’s full 4.5MHz video bandwidth is available. The notch filter is preferable to simple low-pass filters, or to circuit elements which limit bandwidth, since frequencies above the center frequency are still passed. This assures the highest possible bandwidth—and the highest possible video resolution—while still removing NTSC or PAL color signals.

6.6 EXTERNAL PORTS

The DT2851 frame grabber provides two asynchronous data ports which transmit frame data at very high speed to a coprocessor such as the DT2858 auxiliary frame processor. Transfers occur an entire frame at a time; data passes between the frame-store memory buffer on the DT2851 and the frame-store memory buffer on the DT2858.

The data ports use two-way asynchronous handshaking protocols. This means that the rate of data transfer is not controlled by the DT2851 frame grabber’s internal or external sync, but by the DT2858 auxiliary frame processor. This occurs at a rate of about 10 frames per second.

Because the DT2851 frame grabber has two frame-store memory buffers (see Section 6.6), one buffer can display the most recent convolved frame, while the other sends unprocessed frame data to and receives convolved data back from the auxiliary frame processor. When the second processed frame is complete, it can be displayed at the end of the next vertical sync interval, and the first memory can accumulate another convolved frame. The architecture of the DT2851 and DT2858 permits the two boards to perform simultaneous operations on different frames: the DT2851 can be acquiring one frame and displaying a second while the DT2858 performs operations on the third.
PRINCIPLES OF OPERATION

EXTERNAL PORTS

In addition, since the asynchronous data ports effectively constitute an external bus, data transfers between the DT2851 frame grabber and the DT2858 auxiliary frame processor neither tie up the IBM Personal Computer AT bus, nor are limited by the bus’s bandwidth.

6.7 FRAME-STORE MEMORY BUFFERS

Images are acquired to one of the two 256Kbyte frame-store memory buffers. These buffers are dual-ported: they can be accessed by the DT2851 frame grabber during acquisition and display operations; they are also mapped into the IBM Personal Computer AT’s extended memory space, and can be accessed at any time by memory read and write instructions. This permits use of the IBM Personal Computer AT bus to create overlays, to perform frame processing operations, or to store frame buffers on disk. In addition, when acquiring images from a slow-scan device, an entire frame may be acquired in one frame-store memory buffer while the second frame-store memory buffer displays the last complete frame. If only one frame-store memory buffer were available, the display would show the frame being built pixel-by-pixel. Write protection from both the IBM Personal Computer AT bus and the DT2851 frame grabber may be enabled for all bit planes (0 or 1 individually; or 2 and 3 together; or 4 through 7 together), which protects existing overlays or frame data, and allows two 4-bit images to be acquired to the same frame-store memory buffer.

6.8 LUT PROCESSOR

The DT2851 frame grabber contains an LUT processor for real-time image processing. A feedback path connecting the output of the frame-store memory buffer with the input look-up table permits frame data to be passed through any of the eight 256 by 8-bit input look-up tables and returned to either frame-store memory buffer in real-time. Since a variety of operations can be implemented simply by using input look-up tables, the feedback path effectively becomes a processing loop which can perform multiplication or division by a constant; offsetting; AND, OR, and NOR logic operations; and contrast and brightness enhancement on a single 8-bit image. The use of write-protect bits during frame acquisition permits two 4-bit images to be contained in a single frame-store memory buffer. Subsequently input look-up tables
can be programmed to add, subtract, or average the two 4-bit images.

The first frame is acquired using an input look-up table which converts 8-bit linear data into 4-bit logarithmic data, and maps all values to the upper four bit planes of the frame-store memory buffer; before the second frame is acquired, these bit planes are write protected (set bit 3 in OUTCSR to 1); then the second frame is acquired using an input look-up table which maps all values to the lower four bit planes of the frame-store memory buffer. The frame-store memory buffer now contains two images. An input look-up table can be programmed using a nested loop operation to perform Dual Image Pixel Point Processing such as frame addition or subtraction. Passing the frame-store memory buffer containing the two 4-bit images through the look-up table combines the two images, and converts them from logarithmic back to linear values.

6.9 VIDEO OUTPUT

The DT2851 frame grabber contains image display hardware as well. Eight 256 by 24-bit output look-up tables are used to select pseudo-color and intensity for the pixel data. The 24-bit value from the output look-up table is divided into three 8-bit words. These in turn drive the inputs of separate 8-bit digital-to-analog converters, one each for the red, green, and blue output. The data output of each DAC is combined with RS-170 (CCIR for the DT2851-50Hz) sync signals, so it can drive an analog input monitor directly.

Separate sync outputs are provided for use with monitors or cameras which require sync connections. Composite Sync, Horizontal Drive, Vertical Drive, and Composite Blanking signals are made available on separate digital output lines (see Chapter 4).
CHAPTER 1

PRODUCT OVERVIEW

1.1 DT2858 AUXILIARY FRAME PROCESSOR

The DT2858 is a frame processor board designed for high-speed 16-bit image processing with the DT2851 High Resolution Frame Grabber. Each board plugs into one slot of the IBM Personal Computer AT. External ports connect the two boards through a ribbon cable, providing a very high-speed communication path. Used as a special-purpose processor, the DT2858 greatly reduces the time required to accomplish arithmetic-intensive operations on 512 x 512 image frames.

The DT2858 Auxiliary Frame Processor performs the following operations:

- Addition and subtraction of a constant
- Multiplication by a constant
- Arbitrary non-linear transformation of pixel values
- Addition or subtraction of two frames
- AND, OR, or XOR of two frames
- Frame averaging
- Convolutions of arbitrary shape and size
- Division and normalization
- Histogramming
- Zoom (x2, x4, x8), pan, and scroll
PRODUCT OVERVIEW
DT2858 AUXILIARY FRAME PROCESSOR

These operations can be done on one or more frames of data. All operations are carried out and stored with a full 16-bits of precision to prevent loss of accuracy and eliminate round-off errors in intermediate calculations.

The integral arithmetic capability of the DT2858 allows addition, subtraction, multiplication, and division to be done on the data in the DT2858 memory at rates much higher than the IBM Personal Computer AT is capable of, even with an 80287 coprocessor. For example, during a 3 x 3 convolution the ratio of IBM Personal Computer AT to DT2858 processing time is on the order of 250 to 1.

The DT2858 Auxiliary Frame Processor includes the following architectural features:

- Two 8-bit asynchronous data ports for transfer of data frames to and from the DT2851 High Resolution Frame Grabber
- Eight 8-bit to 16-bit look-up conversion tables
- A 16-bit Arithmetic Logic Unit (ALU)
- A 512 by 512 by 16-bit frame-store memory for storage of computational results
- Control logic to normalize or divide 16-bit values into 8-bit results using a successive approximation algorithm
- A 32-bit histogram generator which operates on the low 8-bits of the frame-store memory
PRODUCT OVERVIEW
DT2858 AUXILIARY FRAME PROCESSOR

Figure 1-1 shows a block diagram for the DT2858. In its most simple form, the DT2858 can be thought of as a dual-port memory with integral arithmetic processing. The DT2858 Auxiliary Frame Processor receives 8-bit data from the DT2851 High Resolution Frame Grabber over one of two external I/O ports. These two ports (one for input, one for output) are completely independent of the processor bus and allow entire image frames to be transferred between the DT2851 and the DT2858 at much higher speeds than are possible with the host bus.

1.2 MODES OF OPERATION

There are four user-selectable modes of operation for the DT2858: DATA IN, DATA OUT, NORMALIZE, and HISTOGRAM.

During the DATA IN operation, a frame of 8-bit pixel data is transferred from the DT2851 High Resolution Frame Grabber to the DT2858. Each 8-bit pixel entering the DT2858 is immediately converted to a 16-bit value by the LUT conversion table. This 16-bit value is then used as one input to the 16-bit ALU. The other 16-bit input value to the ALU is read from the DT2858’s frame-store memory. The resulting 16-bit output from the ALU is then written into the DT2858’s frame-store memory.

The on-board frame-store memory stores the results of the arithmetic and logic operations completed by the LUT conversion table and ALU. This 512 Kbyte memory is memory-mapped into the IBM Personal Computer AT extended memory space and is organized as a 512 by 512 by 16-bit block.

The 8-bit pixel data coming from the DT2851 is transferred starting with pixel 0 of its selected frame-store memory and incrementing through each pixel location until 262,144 (256K) pixels have been transferred. X and Y offsets can be programmed on the DT2858, affecting the order in which the data is read from and written to the DT2858 memory. By programming these offsets, neighborhood functions such as convolutions can be performed.

During the DATA OUT operation, a frame of 8-bit data is read from the low bytes of the DT2858 frame-store memory and transferred to the DT2851 High Resolution Frame Grabber. X and Y offsets can be programmed on the DT2858, affecting the order in which the data is read from the DT2858 memory. By programming these offsets, panning and scrolling...
functions can be performed. A zoom function can also be performed. Data in the frame-store memory of the DT2858 is not affected during the DATA CUT operation.

During the NORMALIZE operation, each 16-bit value is read from the frame-store memory and converted to an 8-bit value through comparison with test values stored in a selected LUT conversion table. Each resulting 8-bit value is then stored back into the DT2858 frame-store memory. After this operation, all of the top bytes of the frame-store memory contain zeros.

During the HISTOGRAM operation, the histogram of the 8-bit values contained in the low bytes of the DT2858 frame-store memory is generated and added to the 32-bit values stored in the selected histogram table.
B.1 INTRODUCTION

This appendix contains an alphabetical listing of DT-IRIS subroutines by long-form and short-form names.

B.2 LONG-FORM NAMES AND SHORT-FORM NAMES

<table>
<thead>
<tr>
<th>Long-Form Names</th>
<th>Short-Form Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS_ABORT</td>
<td>ISABRT</td>
</tr>
<tr>
<td>IS_ACQUIRE</td>
<td>ISACQ</td>
</tr>
<tr>
<td>IS_ADD_FRAMES</td>
<td>ISADDF</td>
</tr>
<tr>
<td>IS_ALLOCATE</td>
<td>ISALOC</td>
</tr>
<tr>
<td>IS_AND_CONSTANT</td>
<td>ISANDC</td>
</tr>
<tr>
<td>IS_AND_FRAMES</td>
<td>ISANDF</td>
</tr>
<tr>
<td>IS_CHECK_ACQUISITION_COMPLETE</td>
<td>ISCAQC</td>
</tr>
<tr>
<td>IS_CONFIGURATION</td>
<td>ISCONF</td>
</tr>
<tr>
<td>IS_CONVOLVE</td>
<td>ISCONV</td>
</tr>
<tr>
<td>IS_COPY_REGION</td>
<td>ISCOPYR</td>
</tr>
<tr>
<td>IS_CURSOR</td>
<td>ISCURS</td>
</tr>
<tr>
<td>IS_DEALLOCATE</td>
<td>ISDLOC</td>
</tr>
<tr>
<td>IS_DISPLAY</td>
<td>ISDISP</td>
</tr>
<tr>
<td>IS_DIVIDE_CONSTANT</td>
<td>ISDIVC</td>
</tr>
<tr>
<td>IS_DRAW_ARC</td>
<td>ISARC</td>
</tr>
<tr>
<td>IS_DRAW_LINES</td>
<td>ISLINE</td>
</tr>
<tr>
<td>IS_DRAW_TEXT</td>
<td>ITEXT</td>
</tr>
<tr>
<td>IS_END</td>
<td>ISEND</td>
</tr>
<tr>
<td>IS_ERROR_MESSAGES</td>
<td>ISERRM</td>
</tr>
<tr>
<td>IS_EXCHANGE_FRAMES</td>
<td>ISEXCF</td>
</tr>
<tr>
<td>IS_FILTER</td>
<td>ISFILT</td>
</tr>
<tr>
<td>IS_FRAME_COPY</td>
<td>ISFCOP</td>
</tr>
<tr>
<td>IS_FRAME_CLEAR</td>
<td>ISFCLR</td>
</tr>
<tr>
<td>IS_FREEZE_FRAME</td>
<td>ISFREEZ</td>
</tr>
<tr>
<td>IS_GET_PIXEL</td>
<td>ISGETP</td>
</tr>
<tr>
<td>IS_GET_REGION</td>
<td>ISGETR</td>
</tr>
<tr>
<td>IS_HISTOGRAM</td>
<td>ISHIST</td>
</tr>
<tr>
<td>IS_INITIALIZER</td>
<td>ISINIT</td>
</tr>
<tr>
<td>IS_LOAD_FONT</td>
<td>ISLFNT</td>
</tr>
<tr>
<td>IS_LOAD_ILUT</td>
<td>ISLDIT</td>
</tr>
<tr>
<td>IS_LOAD_ILUT_SVAL</td>
<td>ISLDIV</td>
</tr>
</tbody>
</table>
SUMMARY OF SUBROUTINES

IS_LOAD_MASK
IS_LOAD_Olut
IS_LOAD_Olut_Sval
IS_MULTIPLY_CONSTANT
IS_OFFSET_CONSTANT
IS_SHOW_ATTRIBUTE
IS_OR_CONSTANT
IS_OR_FRAMES
IS_PASSTHRU
IS_PERFORM_FEEDBACK
IS_PUT_PIXEL
IS_PUT_REGION
IS_READ_EPORT
IS_READ_ILUT
IS_READ_ILUT_Sval
IS_READ_Olut
IS_READ_Olut_Sval
IS_REPORT_CURSOR_POSITION
IS_REPORT_DOS_BUFFER_ID
IS_REPORT_ERROR
IS_REPORT_GRAPHIC_POSITION
IS_RESET
IS_RESTORE
IS_SAVE
IS_SELECT_CHANNEL
IS_SELECT_FONT
IS_SELECT_ILUT
IS_SELECT_INPUT_FRAME
IS_SELECT_OUTPUT_FRAME
IS_SET_ACTIVE_REGION
IS_SET_BACKGROUND
IS_SET_CHARACTER_SIZE
IS_SET_CONSTANT
IS_SET_CURSOR_POSITION
IS_SET_FOREGROUND
IS_SET_GRAPHIC_POSITION
IS_SET_MODE
IS_SET_SYNC_SOURCE
IS_SLOW_SCAN
IS_SUBTRACT_FRAMES
IS_SUMMATION
IS_TEST
IS_UNLOAD_FONT
IS_WAIT_ACQUISITION_COMPLETE
IS_WRITE_EPORT
IS_XOR_CONSTANT
IS_XOR_FRAMES
IS_ZOOM_AND_PAN