University of Houston

UNIVERSITY OF HOUSTON • Founded 1927 •

MUG FILE PROJECT REPORT NUMBER UHMUG-15

The UHMFS Computer Software

G. W. Batten, A. Karachievala, and H. H. Nguyen

This project was supported by Grant Number 76-NI-99-012 awarded by the Law Enforcement Assistance Administration, U. S. Department of Justice, under the Omnibus Crime Control and Safe Streets Act of 1968, as amended. Points of view or opinions stated in this document are those of the authors and do not necessarily represent the official position or policies of the U. S. Department of Justice.

MUG FILE PROJECT REPORTS

UHMUG-1 Summary report for a Research Project *"A Man-Computer System for Solution of the Mug File Problem".*
B. T. Rhodes, K. R. Laughery, G. M. Batten, and J. D. Bargainer.

UHMUG-2 *An Analysis of Procedures for Generating Facial Images*
K. R. Laughery, G. C. Duval, and R. H. Fowler.

UHMUG-3 *Factors Affecting Facial Recognition*
K. R. Laughery and R. H. Fowler

UHMUG-4 *The Minolta Montage Synthesizer as a Facial Image Generating Device*
F. H. Duncan and K. R. Laughery

UHMUG-5 *An Analysis of Strategies in Remembering and Generating Faces*
G. C. Duval

UHMUG-6 Data Base No. 1 - *Sketches and Identi-Kit Composites*

UHMUG-7 Data Base No. 2 - *Transcripts of Artist/Technician and Witness Interaction*

UHMUG-8 Data Base No. 3 - *Adjective Descriptors Used in Generating Sketches and Identi-Kit Composites*

UHMUG-9 Data Base No. 4 - *Miscellaneous Data from Sketch and Identi-Kit Generation*

UHMUG-10 *Support Hardware for Image Analysis Techniques Applied to the Mug File Program*
J. D. Bargainer

UHMUG-11 *Forgery Application of a Pattern Recognition Algorithm for Facial Images*
B. T. Rhodes and K. Prasertchuang

UHMUG-12 *An Evaluation of the UHMFS Facial Image Pattern Recognition Algorithms*
B. T. Rhodes and C. R. Walters

UHMUG-13 *FORTRAN Subroutines for the Pattern Recognition Algorithm Designed to Find "Look-Alikes" in a Mug File*
K. Sumney

UHMUG-14 *A Computer Simulation of the Minolta Montage Synthesizer*
G. W. Batten and T. Wiederhold

UHMUG-15 *The UHMFS Computer Software*
G. W. Batten, A. Karachievala and H. H. Nguyen

UHMUG-16 *Miscellaneous Computer Software for the Mug File Project*
G. W. Batten

MUG FILE PROJECT REPORT UHMUG - 15

The UHMFS Computer Software
Part 1

by

George W. Batten, Jr.
Asif A. Karachiwala
Ha Hoang Nguyen

University of Houston
Houston, Texas

October 1, 1977

This project was supported by grant number 76-NI-99-012 awarded by the Law Enforcement Assistance Administration, U. S. Department of Justice, under the Omnibus Crime Control and Safe Streets Act of 1968, as amended. Points of view or opinions stated in this document are those of the authors and do not necessarily represent official position of the U. S. Department of Justice.

Table of Contents

Table of Figures

CHAPTER 1

INTRODUCTION

This report describes computer software for the University of Houston Mug File System (UHMFS). The intended use of this system in law enforcement is described in the summary report UHMUG-1. The present report, which is intended for readers with a background in the technical aspects of computer software, provides detailed information on the internal structure of the UHMFS software. It is divided into three parts, the first covering the design of the software, the second being a User's manual, and the third containing detailed documentation.

The major objective in development of this software was to combine a new pattern recognition technique with a state-of-the-art law enforcement information retrieval system. The system would provide enhanced capability in criminal investigations by allowing investigators to include facial measurements obtained from photographs or witness sketches as parameters for searching known-offender files by computer.

An algorithm for pattern recognition in such a system was developed by James R. Townes at the University of Houston. That algorithm has been implemented in software described in report UHMUG-13, which includes program listings. The present report on the information retrieval software describes the file structure and program linkage to the pattern recognition subroutines, but it avoids direct reference to the specific algorithm used since it is likely that improved algorithms will be available shortly.

The CRIME system of the Oakland Police Department (California) was selected as the existing system most consistent with the objectives for UHMFS. Experience with that system revealed appreciable success, as indicated by the Oakland Police Department CRIME File System Project Report [1], but that report also pointed out certain problems. The University of Houston research group decided to modify the CRIME system software to correct some of its defficiencies and to adapt it for the pattern recognition algorithm. The resulting UHMFS software resembles the CRIME software closely.

One major problem with the CRIME software concerns the multiterminal operation. Ideally, investigators on different terminals would be unaware of the actions of one another, but in the CRIME system, an active user at any one terminal locks out users at all other terminals. The available computer capability is hardly used, however, since most of the time the computer is waiting on human responses. The UHMFS software includes programs which allow concurrent multiterminal operation except when the computer is actively searching the data base. System response is such that most of the time an investigator feels that he has complete control of the system.

Another major problem of the CRIME system is the inefficiency of data-base revision (updating). A completely new updating program has been devised for UHMFS. The new program requires much less time for updating, an important operational factor since the data base cannot be queried during updating.

The data base of both systems (UHMFS and CRIME) is divided into two parts, the Subject File (SF) and the Vehicle File (VF), with related files (reference files and key files) which aid in fast retrieval using frequently-used parameters. Inclusion of the facial measurements in the Subject File required modification of the CRIME system file structure; some data-compression techniques were used so that the size of records in the Subject File of UHMFS is the same as that in the CRIME system even though the records can contain up to twenty numbers relating to image measurements. Thus, UHMFS will operate on the hardware configuration of the CRIME system.

The software of UHMFS comprises special application programs operating under the Hewlett-Packard Disc Operating System (DOS-III) on the Hewlett-Packard 2100A computer. These programs are under conτrol of the system operator at the system console. There are four main programs. Thay are:

INTIL for initializing the data base,

UPDAT for updating the data base files,

EDIT for lising and purging records in the files,

QUERY for searching the data base and retrieving records.

The program INITL, UPDAT, and EDIT can be used only from the system console. Program QUERY is activated and deactivated from the system console, but it accepts search commands from and sends results to auxiliary (remote) terminals.

CHAPTER 2

FACIAL MEASUREMENTS

2.1 Introduction

It seems obvious (some existing systems to the contrary) that any law enforcement data retrieval system should allow an investigator considerable freedom in specifying criminal characteristics (search parameters) for searching the data base. Thus, for example, estimates of a criminal's height by witnesses are often in error, so an investigator might choose to specify a range rather than a single height. The same considerations apply to the use of specific identification techniques such as fingerprinting and mug-shot pattern recognition, the subject of this report. The extent to which this is possible in UHMFS can be seen by examining the user's guide in Part II of this report. Here we present a brief description of the user's view of the system as it relates to the mug file problem so that the reader will have a better understanding of the software.

2.2 The Data Base

Each of the two parts of the data base of UHMFS contains a master file: file SUBJF for the Subject File, and MVEHF for the Vehicle File. Each record of a master file contains all of the known (to the computer system) information on a single known-offender (SUBJF) of vehicle (MVEHF). Thus, each record of SUBJF contains typical physical descriptors (sex, height, weight, hair color, etc.), fingerprint codes, addresses (locations) of mug shots in a mug file, and measurements of facial characteristics taken from the mug shots. The latter is a unique feature of UHMFS; it permits sorting of suspects on the basis of facial characteristics. The records of the Vehicle

File contain information on vehicles which have been encountered in normal police activity. This file has only incidental importance to the present chapter.

In UHMFS, the Subject and Vehicle Files are used independently of each other, but this use involves common principles of data-base management and information retrieval. Briefly, there must be some method of adding new information to these files, some method of purging out-of-date information from them, and some method of finding all those records which relate in some way to the clues for a particular crime. The adding and purging of information are referred to collectively as "data-base management," and the recovery of files containing information based on certain key data is referred to as "information retrieval." Data base management in UHMFS is the topic of Chapter 3. Information retrieval operations, which are done by program QUERY, follow the pattern shown in the following section.

2.3 Querying the Subject File

Assume that the user has been through the log-on procedure and that he has selected the Subject File (SF) by the SF command. For the sake of example, we shall assume that the criminal being sought is a white male, 23-27 years old, 69-72 inches tall, with red hair and a cauliflower ear; and that the crime is auto theft. The investigator would type the following (note that the character in the first column is the prompt character issued by the computer and that user responses are underlined)

```
↑SEARCH       (tells the system that search parameters follow)
↑A1           (first search parameter;  subject is male)
```

```
↑B1           (second search parameter; subject is white)
↑C1           (third search parameter; age between
#23,27                                           23 and 27)
↑F1           (fourth search parameter: height between
#69,72                                       69 and 72 inches)
↑H4           (fifth search parameter: red hair)
↑M1           (sixth search parameter: cauliflower ear)
↑E9           (seventh search parameter: crime is auto theft)
↑DONE         (indicates the end of the search-parameter list)
```

At this point, the system searches through the data base and locates all records matching all of the search parameters. Then it presents to the user

```
THERE ARE nnnn 'HITS'
DO YOU WANT TO USE THE LOOK-ALIKE ALGORITHM?  (YES OR NO)
↑
```

to which the user must respond. If he responds with NO, he can specify more search parameters, or have the hit list printed or displayed. If he answers YES, the computer checks to see if facial measurements have already been supplied (i.e., via code Z1 when the search parameters are being typed). If so, it skips the next step; otherwise it requests them by typing

```
ENTER FACIAL DATA TYPE (PHOTO, SKETCH, OR COMPOISTE)
↑
```

to which the user gives an appropriate answer. The computer responds

```
ENTER FACIAL MEASUREMENTS, ONE AT A TIME AS ASKED FOR
#1:
```

thereby requesting facial data. After the user has entered all of the measurements as required, the computer sorts the hit list into order so that the "best look-alike" is first, then it types

```
THE 'HITS' ARE NOW ORDERED ACCORDING TO THEIR FACIAL DISTANCE
↑
```

at which point the user can request the system to print or display the list, if he wishes (the option of specifying other search parameters is available, but it is faster to do all of the searching before ordering the hit list).

2.4 File structure: UHMFS vs. CRIME

From the user's viewpoint, UHMFS is very similar to the CRIME system from which it was developed. Internally, however, there are some important differences relating to the structure of the database files. As the file structure is described in detail in Part 2 of this report, we shall be content at this point to give an overview of it.

The Vehicle file in UHMFS is identical to the Vehicle File in the Crime system, since this file has nothing to do with facial pattern recognition, we shall not consider it here.

The record format of the master file (SUBJF) of the subject file has been changed substantially. The name and nickname fields have been compressed by taking advantage of the fact that there are only 31 characters (the letters A-Z, period, comma, hyphen, space, and apostrophe) which appear in these fields. Rearrangement of other fields has provided further compression of the records. As a result, it has been possible to compress the data so that the records of UHMFS contain all of the information of those of the CRIME system plus twenty eight-bit numbers (i.e., numbers between 0 and 257, inclusive), yet the record lengths of the two systems are the same. The two record formats are shown in Figure 2.1.

Bit number

Word No.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	ID Number (Encoded)															
2																
3	Name (24 ASCII characters)															
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15	Nickname (10 ASCII characters)															
16																
17																
18																
19																
20	▨	▨	▨	▨	District Code				Year of Arrest							
21	Mug Shot Address #1															
22																
23	" " " #2															
24																
25	" " " #3															
26																
27	" " " #4															
28																
29	Fingerprint Address															
30																
31	▨	▨	▨	▨	Finger 0 Code				Code #2				Code #3			
32	▨	▨	▨	▨	1 " #1				"				"			
33	▨	▨	▨	▨	2 "				"				"			
34	▨	▨	▨	▨	3 "				"				"			
35	▨	▨	▨	▨	4 "				"				"			
36	▨	▨	▨	▨	5 "				"				"			
37	▨	▨	▨	▨	6 "				"				"			
38	▨	▨	▨	▨	7 "				"				"			
39	▨	▨	▨	▨	8 "				"				"			
40	▨	▨	▨	▨	9 "				"				"			
41	Sex	▨	▨	▨	Race				Year of Birth							
42	Type of Crime 2								Type of Crime 1							
43	Height							Weight								
44	Hair Color				Hair Style 3				Hair Style 2				Hair Style 1			
45	Eye Color				Eye Defects				Ears				Lips			
46	Complexion				Tattoo Marks				Facial Hair				Teeth			
47	▨	▨	▨	▨	Amps & Deform				Visible Scars				Peculiarities			
48	▨	▨	▨	▨	▨	▨	▨	▨	Speech 2				Speech 1			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

CRIME

Bit number

Word No.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	ID Number (Encoded)															
2																
3	▨	Name (24 compressed ASCII characters)														
4	▨															
5	▨															
6	▨															
7	▨															
8	▨															
9	▨															
10	▨															
11	▨	Nickname (10 compressed ASCII characters)														
12	▨															
13	▨															
14	▨	▨	▨	▨	Year of Birth											
15	Mug Shot Address #1															
16																
17	" " " #2															
18																
19	" " " #3															
20																
21	" " " #4															
22																
23	Fingerprint Address															
24																
25	Hairstyle 1				Finger 0 Code #1				Code #2				Code #3			
26	" 2				" 1 "				"				"			
27	" 3				" 2 "				"				"			
28	Hair Color				" 3 "				"				"			
29	Amps & Deform				" 4 "				"				"			
30	Scars				" 5 "				"				"			
31	Tattoo Marks				" 6 "				"				"			
32	Peculiarities				" 7 "				"				"			
33	Speech #1				" 8 "				"				"			
34	" #2				" 9 "				"				"			
35	Eye Clr		Eye Def		District Code					Year of Arrest						
36	Sex	Lips			Ears			Teeth			Race			Facial Hair		
37	Type of Crime 2						Type of Crime 1						Complexion			
38	Height							Weight								
39	Facial Meas Set 2: Measurement 1								Facial Meas Set 1: Measurement 1							
40	" " 2								" " 2							
41	" " 3								" " 3							
42	" " 4								" " 4							
43	" " 5								" " 5							
44	" " 6								" " 6							
45	" " 7								" " 7							
46	" " 8								" " 8							
47	" " 9								" " 9							
48	" " 10								" " 10							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Note: ▨ = Not used.

UHMFS

SF Master File Record Structure

Figure 2.1

2.5 Modifications

Since the Subject File structure was modified as shown in Figure 2.1, adaptation of the existing CRIME system software to UHMFS required modification of all three of the main programs INITL, UPDAT, and QUERY.

The main changes made to program INITL, which initializes the files of the data base, were related to redefining the record structure of the Subject File. Specifically, files FIEDD and SUBDE, which provide file structure definitions to the software, were changed to account for the new record organization in SUBJF. Program INITL also creates the Master Interface File (MIF), whose length has been extended to 106 words to provide room for facial measurements.

The structure and operation of program UPDAT has been completely changed from that of the CRIME system. The new program is discussed in Chapter 3 of this report.

The major modification of program QUERY centered on development of an additional overlay segment to handle the look-alike algorithm. That segment, QSEG4, orders the subjects in the hit-list according to the distance between them and the target as determined from user-supplied measurements of a photo, sketch, or composite image. Other modifications to the program appear in the following list which summarizes the look-alike-algorithm-related modifications to QUERY:

* addition of overlay segment QSEG4 to implement the look-alike algorithm;
* addition of various query commands for control of the look-alike algorithm;
* change of the COMMON area to accommodate additional parameters;

- insertion of calls to subroutine EXPND at appropriate places for expanding the compressed name and nickname fields back into ASCII form;
- modification of bit masks and positions of the various data fields in the records of SUBJF wherever needed.

The modifications listed above do not include the modifications necessary to implement concurrent multiterminal operation, which is the subject of Chapter 4 of this report.

CHAPTER 3

DATA BASE MAINTENANCE

3.1 Introduction

In this report the term data-base maintenance refers to the process of revising the files of the data base to keep the information in them current. Three basic operations are involved in this process: deletion of out-of-date records, insertion of new records, and correction of certain fields in old records. This chapter is concerned with the computer programs which handle this process. One of these programs (EDIT) has in addition, the ability to list contents of files.

In the CRIME system, all data-base maintenance operations were done by program UPDAT. That program suffered from poor choice of strategy, so that the updating process was very time consuming. This caused operational difficulties in the system's use. The problem was aggravated by the feeling of the system designers and users that, because of the complexity of the updating process, it was necessary to verify the updated file structure by using program VERUP, also a very time consuming program.

In UHMFS, these problems have been relieved to a considerable extent by using a completely new data-base maintenance system. This system comprises four parts: the data input program (or subprogram), the master-file updating program (MERGE), the key- and reference-file rebuilding program (RBLD), and the editing and listing program (EDIT). Data input can be done via the stand-alone program UPDAT, or via program QUERY in the time sharing mode. In either case, the process is under the control of the system operator at the system console.

Program EDIT is a stand-alone program for deleting individual records, listing file contents, and making limited changes in the contents of individual records.

3.2 The file structure

Each master file (SUBJF or MVEHF) has an associated set of reference and key files. These files, which are used for rapid retrieval of records based on certain frequently-encountered search parameters, are related as shown in Figure 3.1.

We shall use the term key to refer to a number obtained by combining particular search parameters in a certain way. For example, a key corresponding to the reference file SRA is determined by the formula

$$K_{SRA} = 6 \cdot S + R,$$

where S is the sex (0 for female, 1 for male), and R is the race (0 for white, ..., 4 for undetermined). Each reference file has an associated set of search parameters which are used to form keys for this reference file (it may also have an associated search parameter which is not used in forming keys). Thus, corresponding to each record in the master file there is a specific key associated with a given reference file.

The reference files form an index system for locating information in the master file (i.e., we have an "inverted list" structure). Each reference file record contains a pointer to the corresponding master-file record. The records in a reference file are sorted into ascending order of keys for this file, but the keys are not stored in the reference file. Instead, they are determined by pointers in the corresponding key file. With two exceptions, each record in a key file comprises only one word, a pointer to a record in

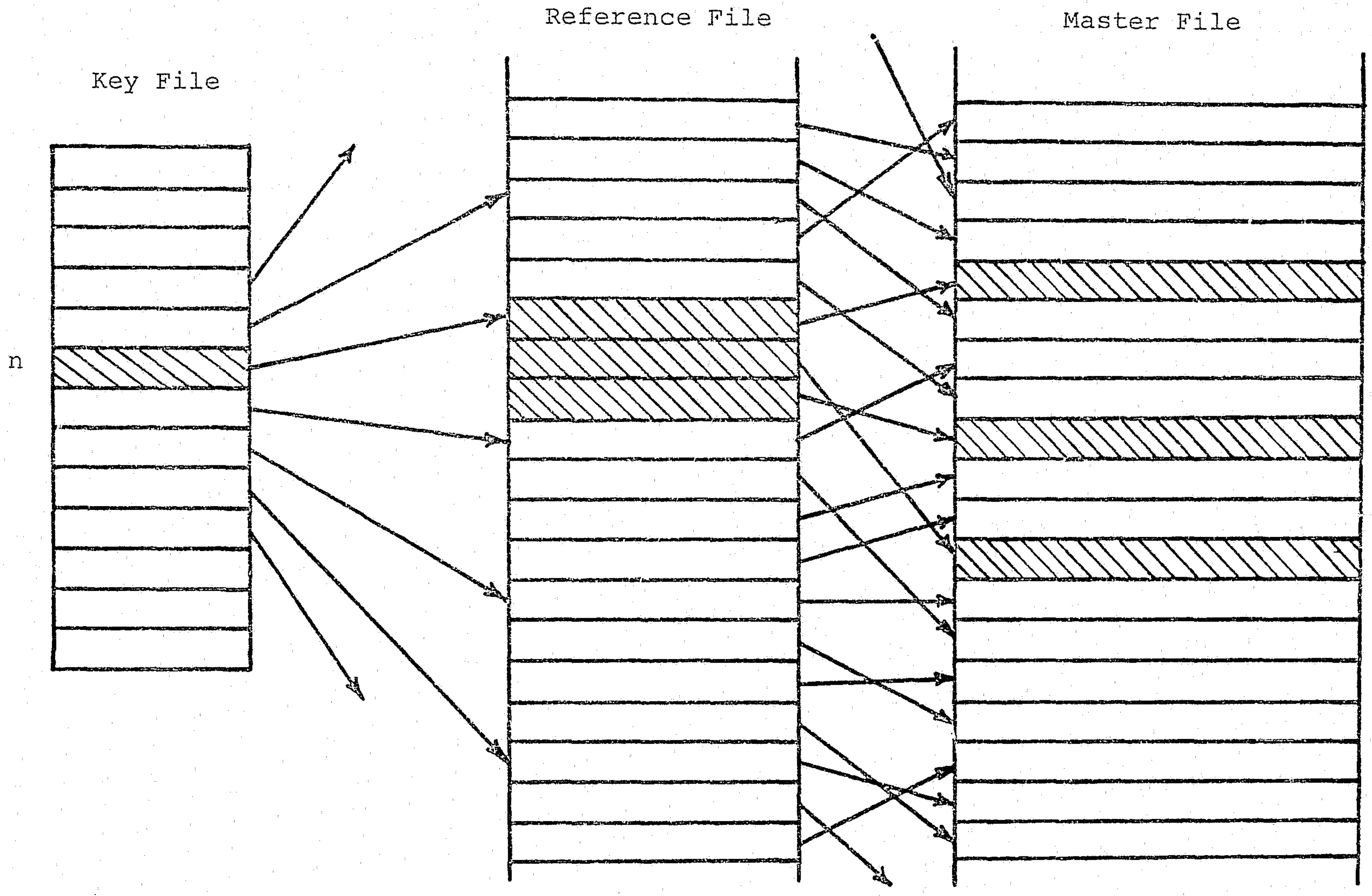


Figure 3.1

Relations Between Files

the reference file. Record number n of the key file points to the first record whose key value is greater than or equal to n. Thus, master-file records whose key is n can be found by using the pointers of those records of the reference file corresponding to record n of the key file. The situation is illustrated by the shaded records in Figure 3.1.

The two exceptional records in a key file are the first record and possibly the last few records. Since the key-file pointer for record 1 will always be 1, this record is used to store the length of the reference (and master) file. Zero in any key-file record indicates that there are no master- or reference-file records corresponding to this or any subsequent key-file records.

3.3 Strategy for File Updating

File updating comprises three parts: data input, master-file updating, and key- and reference-file rebuilding. This organization differs from that of the CRIME system, in which the three operations are done concurrently, and the key- and reference-files are modified rather than being rebuilt. An overall flow chart for the process is shown in Figure 3.2.

In the data-input phase, the system reads data cards, checks them for detectable errors, and writes them to the Master Update File (MUF). Checking is, at this point, made on a single card basis; no check is made to determine if cards or fields of information are duplicated (as might occur, for example, if there are multiple corrections to a particular record of a master file). Any input records found to be in error are reported on the system console and omitted from the Master Update File. When the last input card has

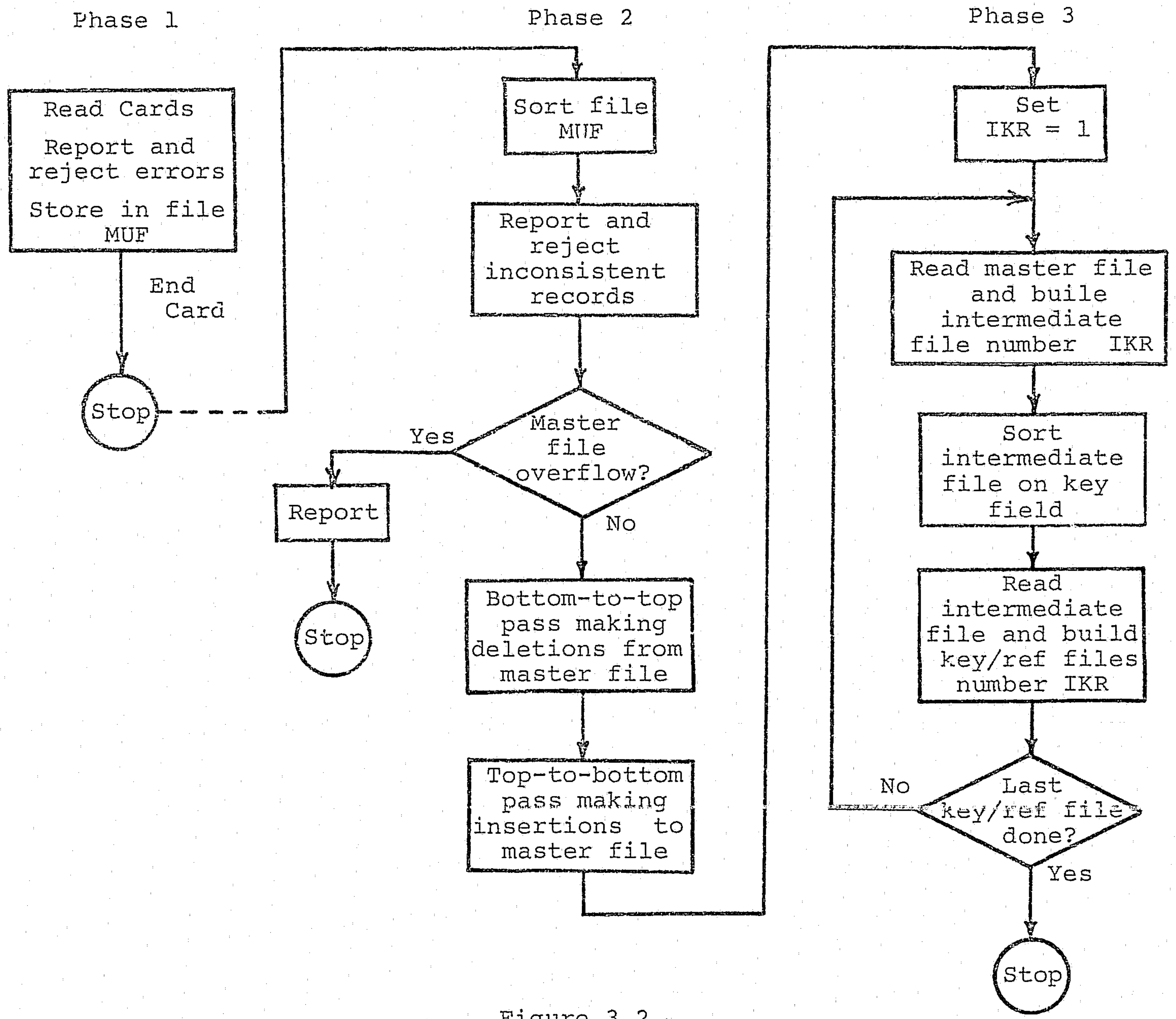

Phase 1
Phase 2
Phase 3
Read Cards
Report and reject errors
Store in file MUF
End Card
Stop
Sort file MUF
Report and reject inconsistent records
Master file overflow?
Yes
No
Report
Stop
Bottom-to-top pass making deletions from master file
Top-to-bottom pass making insertions to master file
Set IKR = 1
Read master file and buile intermediate file number IKR
Sort intermediate file on key field
Read intermediate file and build key/ref files number IKR
Last key/ref file done?
No
Yes
Stop

Figure 3.2

Flowchart of Updating Subsystem

been read, the system operator is notified and the input phase is terminated. The next phases are not started automatically since those phases require that QUERY operation be shut down until completion of updating, a relatively lengthy process, and there may be some important QUERY jobs to be done first. Note that no changes to the system data base have been made at this point.

Master-file updating comprises two parts. In the first part, entries in the Master Update File (MUF) are sorted into the order of the corresponding records in the master files. The records are checked for conflicting duplicate records, and any found are reported and eliminated. Master-file sizes are checked for possible overflow; if overflow is detected, it is reported and updating operation is terminated (the data base has not been changed at this point; records must be deleted from the data base to continue updating). Finally, in the second part of this phase, the sorted input records are merged into the master files, requested deletions being made in the process. Note that this last operation is done using two passes through the master file, one bottom-to-top pass for deletions and one top-to-bottom pass for insertions. This completes phase two; since only the master files have been revised, QUERY operations cannot be done at this time. In view of this fact, it seems pointless to interrupt the updating process at this point. Therefore, program MERGE, which performs the phase two operations, initiates operation of phase three automatically.

Phase three constructs the key- and reference-file pairs one at a time using multiple passes through the master files. For each key- and reference-file pair, the program (RBLD) constructs an intermediate file which is just the reference file with the key for each record appended to that record. This intermediate file is sorted into ascending order on the keys, which is, of course, the order of the records in the reference file. The final step for this phase is the copying of records from the intermediate file into the reference file (keys are removed, of course) and construction of the key file; this is done in one pass through the intermediate file.

Since the key and reference files are reconstructed each time the data base is revised, hardware errors occurring during the construction of these files do not propogate into future revisions. Thus, with UHMFS it is unnecessary to perform key- and reference-file verification such as is done by program VERUP in the CRIME system. This saves a large amount of time. UHMFS updating involves less manipulation of the records in the master files; we feel that this makes it more reliable than that of the CRIME system.

Updating involves a large number of sorts of long files, so speed of the procedure depends on the efficiency of the sorting algorithm used. In UHMFS, all sorting in the updating process is done by the method of polyphase merging, a technique which is well suited to sorting large files on mass storage devices (Knuth [2, p.266]). Usually the sorting time is less than the time required for the multiple passes through the master file, most of which occur in the construction of the key and reference files. Some time could be saved by reorganizing the last phase to construct several key- and reference-file pairs in a single pass through a master file. In doing so, it would be important to consider the availability of mass-storage space for the intermediate files.

3.4 Other operations

Program EDIT can be used for deletion and limited correction of records in the master files. Deletion of records is done by replacing their name fields with --DELETED--, and by setting all other fields to zero. The records are not actually removed from the master file at this time, but they are removed when the data base is revised by the updating program. Correction of records is limited to fields which do not affect any keys of the reference files. These limitations make such editing fast since it is unnecessary to rebuild the key and reference files after editing a record. Corrections which involve other fields can be done via the updating system, of course.

CHAPTER 4

CONCURRENT MULTITERMINAL OPERATION

4.1 Introduction

The idea of incorporating a concurrent multiterminal (time-sharing) capability in UHMFS is based on the fact that, since human thinking processes and responses are slow relative to the logical and arithmetic capabilities of the computer, it is possible to switch the computing resources from one user to another in such a way that each user can interact with a terminal online to the computer as if he has sole access to the computer. The same considerations can be applied to peripherial I/O devices attached to the computer, which have a speed disadvantage compared to the central processing unit. Thus it is natural to have the I/O devices share the CPU sequentially in time, and to consider the users to be part of these devices.

There are, of course,many general purpose time sharing systems in existence, but the scope of the present project precluded the use of one of these. Since a major portion of the information retrieval software(the CRIME system software), was already available, it was impractical to build a complete time-sharing file maintenance and retrieval system from the ground up. Thus, the only reasonable approach was adaptation of the CRIME system software for time-shared operation of the QUERY portion. For several reasons it was desirable to avoid modification of the operating system software (HP DOS-III). The result is a special purpose time sharing system for the QUERY portion of UHMFS. All coding relating to the time-sharing aspects of the system is in the Fortran language. Note that file updaing is done by a separate program (UPDAT) which does not operate in the time-sharing mode.

In time-shared operation, there may be several users of the QUERY program at the same time. One method for such operation is to provide separate copies of the program for each user. However, considering the limited computer memory of 24K words and the sizable QUERY program, an alternative method is needed. Hence, the design selected is to have one program which is shared, but is a separate process for each user; the processes run concurrently. By concurrent, we mean that two or more processes are in a state-of-execution at the same time. A process is in a state-of-execution if it has been started but not completed or terminated. Such concurrent execution of two or more processes, which is called multiprogramming, is employed in this project.

A single copy of the program which can be used concurrently by several processes is called a pure procedure or is said to be reentrant. For a program to be reentrant it must not modify itself, and it must not store data local to itself. Hence separate data and temporary storage areas must be provided for each user of the program. This was taken care of in the context-block as explained in the next paragraph.

In order to switch the physical processor from one process to another, some information must be saved when a process is removed from control, and restored again when a process returns to control. This information is often called the context-block. The following is the type of information that must be saved and restored.

- The process must know what instruction to execute next when it assumes control of the physical processor.

- The address space of the process must be saved. This also ensures separate data and temporary storage areas as required for reentrant programs, mentioned in the previous paragraph.
- The state of I/O devices affecting the process must be saved.

Other systems may require additional information, but this is sufficient for QUERY.

Assignment of the physical processor to processes is scheduled by processor management, a general description of which is illustrated in Figure 4.1. The process-scheduler and the traffic-controller are the two modules of the processor-manager that control and keep track of state transitions of different processes. The process-scheduler determines which of the processes receives the processor and at what time. The traffic controller keeps track of the status of each process. When a terminal user signs-on, his process is assigned the READY state. Next, the process-controller in conjunction with the traffic-controller assigns it to the physical processor and labels its state as RUNNING. While it is running the traffic controller continuously updates the status information on other concurrent processes. When the process requests an I/O operation, it is put into the WAIT state until the I/O request is complete, and then it is assigned the READY state again. Each process has an identical state diagram. It is worthwhile to note here, that all the processes are identical since they all execute the same program. As will be discussed later, no time-slice allotment was employed, so there is no direct path from the RUNNING state to the READY state.

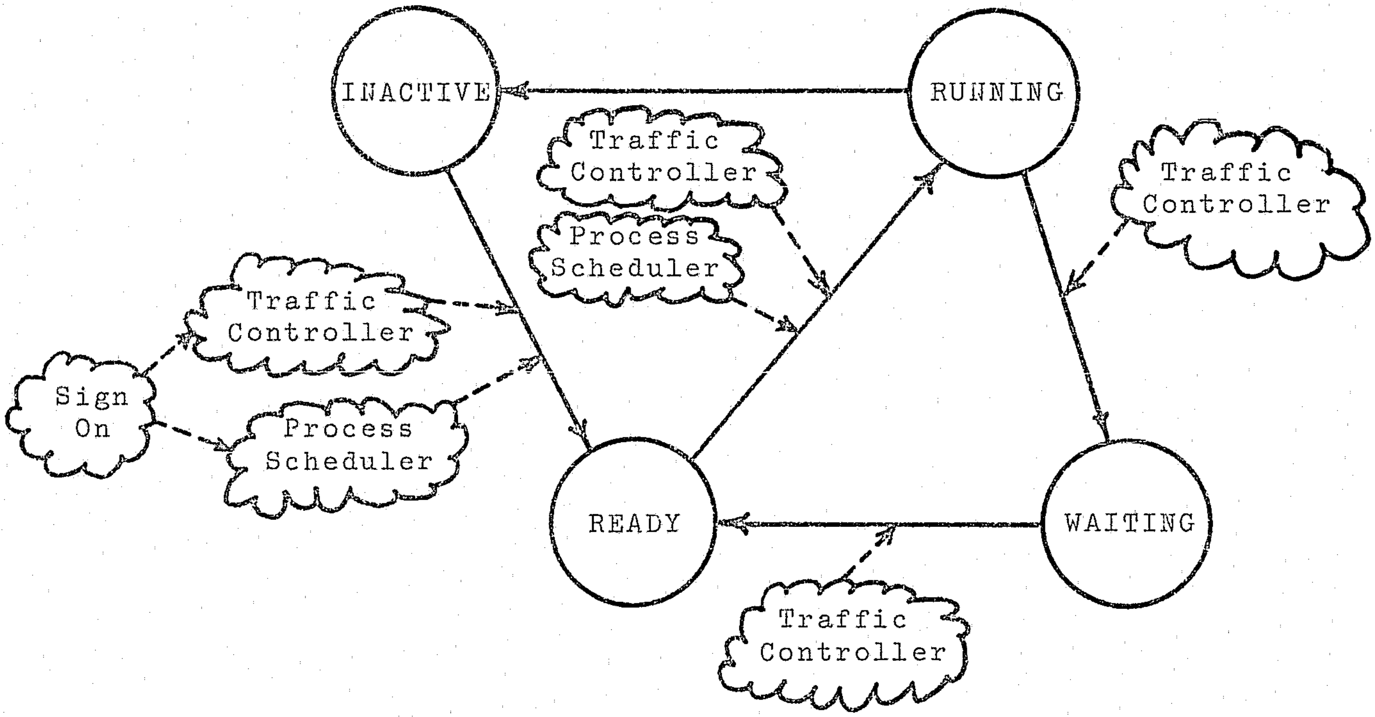


Figure 4.1

Process-Manager State Diagram

4.2 Design Implementation

The overall flow chart for the segments of the original QUERY program is shown in Figure 4.2. In brief, the functions of each of these segments are as follows.

- QUERY: This is the main program to which control is given by DOS directive from system console to initiate the QUERY program. It is a dummy main program for loading purposes and establishes a common block of 128 words for use throughout QUERY.
- QSEG1: This overlay segment initiates operator communication through the system console, checks the validity of disc packs to be used, and verifies the active terminals by L.U.N. (logical unit number) and initializes certain common buffers and flags.
- QSG1A: This overlay segment polls all terminals for attention to sign on, and once a terminal has signed on, it transfers control to QSG1B for query commands.
- QSG1B: This overlay segment is responsible for controlling the operation of the query functions and does all user communications. It accepts and performs the various query commands entered at the terminal. Prior to doing any I/O or EXEC calls, it checks to see if other terminals are requesting attention.
- QSEG2 & QSEG3: These two overlay segments actually handle the search based on the characteristics specified. In the time-sharing system these run to completion before releasing control, therefore these do not have appreciable effect on the object of incorporating time-sharing in the system and they need not be further elaborated.

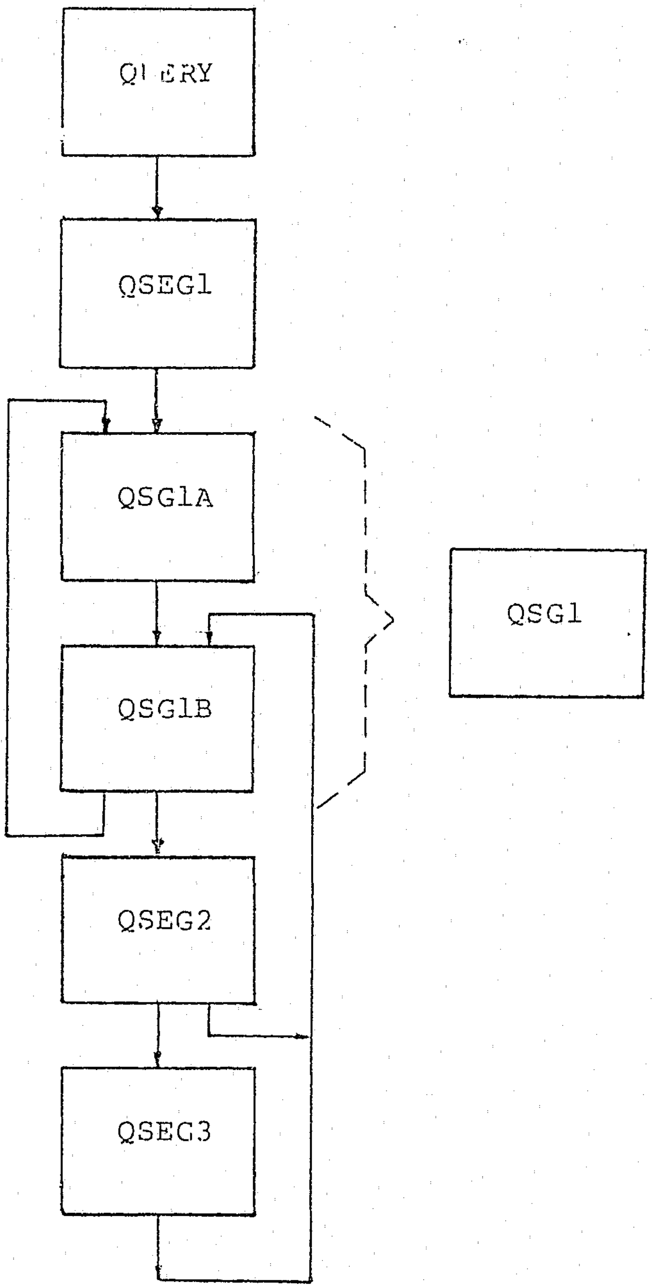


Figure 4.2

Flowchart of Query Subsystem

Since segments QSG1A and QSG1B handle the terminal sign on procedure and user communications, these have been modified in order to implement time-sharing. Minor modifications have been made in QUERY and QSEG1.

Segment QSG1A polls the terminals for attention to sign on, and transfers control to QSG1B when a terminal signs on. It is very possible that after a process for one terminal has reached QSG1B, some other terminal would want to sign on. In this case, switching control from one terminal to another would require jumping back to segment QSG1A, and hence would require swapping of overlays between main memory and disc storage. This would also require files to be opened and reset, and certain buffers to be initialised in each segment each time they were swapped. Hence to prevent this unnecessary overhead, it was necessary to combine QSG1A and QSG1B into one overlay segment QSG1. Since segment QSG1A was comparatively small, the combined segment QSG1 did not overflow the memory.

Now let us discuss the information management required for multiprogramming QUERY; i.e., the context block for each process. In the original version of QUERY, all the variables which were modified during the execution of QUERY, were assigned a common block of 128 words. However, there were some unused locations in this block. These came in handy for assignment to certain variables local to QSG1, such as loop variables and flag variables. The idea was to maintain a copy of this common block for each process and swap them whenever control was switched from one process to another. Hence, it was necessary to store all the variables that were manipulated during QSG1, in this common block.

A second common block was inserted for storing certain status variables of the terminals and the process scheduling queue, and for buffers in which information read from different terminals could be stored. Also included in this common block was a 128 x 6 word array for storing a copy of the first common block for each terminal. Detailed descriptions of all these variables and arrays is given in Part 3 of this report. Brief mention has been made here to indicate the division of the information storage area into two blocks, one which is swapped each time the processes are switched, and the other which maintains the status of each process and can be modified or tested during execution of any process.

The characteristics that are entered during execution of QSG1 are stored in the master interface file MIF when control is passed on to the search routines; search is performed based on parameters in this file. The hits encountered in the search are passed back to QSG1 in a master hit file MHF. There are also six hit files HIT1 to HIT6, one for each terminal, for saving the hits for future reference once that terminal user given the END command to end his query job. With the inclusion of multiprogramming, search parameters can be concurrently entered from many terminals, so a separate interface file was created for each terminal - MIF1 to MIF6. Each of these is 96 words long. Since hits obtained during search routines also need to be saved for each terminal, information in MHF on return to QSG1 from search routines, is immediately transferred to the appropriate hit file HIT#. In the original system, file HIT1 was 8192 words long while HIT2 to HIT 6 were 256 words long. These sizes are maintained in UHMFS, but if the number of hits in MHF is greater than the size of file HIT#, a warning to that effect is printed on the appropriate terminal so that the investigator can rerun the search with more parameters, thereby reducing the number of hits. These sizes of hit files are considered suitable, but it would not be difficult to increase the sizes of these files.

Multiterminal operation within the DOS-III operating system is made possible by the fact that I/O devices can operate concurrently provided they are assigned separate buffer areas. Completion of an I/O operation is detected by polling the device via an operating-system executive call. Thus, conversation between the system and a terminal user is in the following sequence:

1. the process for a terminal issues an output operation to that terminal; this may be in the form of a question (often just a prompt character), a comment, or some information from the data base;
2. the system puts the process for this terminal into the WAIT state and proceeds to the next process in the READY state;
3. the user responds with appropriate information terminated by a carriage return;
4. the system, which continually polls the terminal for completion, detects completion of the input operation for this terminal and it puts the corresponding process into the READY state;
5. when the system puts the process for this terminal into the RUNNING state, it processes the input, prepares a reply, and goes to step 1 to send the reply to the user.

Note that each process follows this same sequence, and that all active terminals are polled in steps 1 and 5.

Thus, the system response time as seen by a particular user depends almost solely on the time the various processes spend in step 5 (other steps require very little time). Since, however, certain tasks in step 5 must be completed before the I/O devices are polled, there will be times (such as during an active search) when response time will be appreciable. There are two routines which are important to the time-sharing operation, the process scheduler READA and the traffic controller POLL. Now let us consider the significance of these.

QSG1 is divided into 29 program steps i.e. there are 29 entry points. These entry points are given statement numbers 5201, 5205, ..., 5229. Whenever the program reaches a point where it needs to read from the terminal, READA is called. The general pattern of coding at this point is:

```
                NSTMT  ← ##
                CALL READA
                GO TO 5200
        52##    ..........
```

Here NSTMT is the next statement number for the current process, that is passed on to READA. On exit from READA, NSTMT holds the next statement number for the new process. Statement number 52-0 is:

```
        5200 GO TO (5201,5202,............,5229)NSTMT
```

Hence, on exit from READA, control will be transferred to appropriate entry point for the new process. A similar coding pattern is also employed where PRINT directive from the terminal user is processed, to interleave printing of the entries in 'hit files'. Traffic controller is called from within READA and also at several places in the QSG1 program. It updates the status information of various terminals and the scheduling queue.

Each of the terminals has two status flags assigned to it, they are IOCMND and IOSTAT. IOCMND, when set, indicates that I/O READ EXEC routine is to be called for that terminal when that I/O device becomes free. While IOSTAT, when set, indicates that the process is ready to be scheduled for running, i.e. ready to be included in the scheduling queue when the I/O device becomes free. These two flag variables are employed to provide concurrent I/O on the terminals. The object was to be able to replace all Fortran READ and WRITE statements by I/O EXEC calls which would initiate the I/O and then without waiting for completion, transfer control to the next instruction in the program. The flow charts for READA and POLL are shown in Figures 4.3 and 4.4 In conjunction with the above discussion they become quite self explanatory. There are however a couple of things that may need elaboration. The KOMON area mentioned in Figure 4.3 is the common block which is specific to each process and is swapped each time processes are switched. Each terminal is assigned a buffer of 36 words into which information from that terminal is read. Before exit is made from READA, the contents of the appropriate buffer are transferred to a local buffer of 36 words in the KOMON area.

Since the time usually required for searching is not intolerably long, it was decided to allow searching to proceed uninterrupted until completion. Thus, during searching (i.e., when QSEG2 or QSEG3 is in control) there is no system—user conversation. Consider a case where one user has given the PRINT command, and while the hit list is being printed on his terminal, some other user requests a search. Since printing is interleaved, it is possible that in between printing of two lines on one terminal, the process for another terminal may enter a search routine, causing noticable time lag between two lines. This was considered undesirable and hence PRINT was given priority by introducing the feedback loop in READA, as shown in Figure 4.3. This can be removed if the other priority arrangement is more desirable.

It is worth noting that this part of the system was designed and implemented in the Fortran IV language on DOS-III, a system which was not particularly designed for time-sharing operation.

References

[1] Oakland Police Department CRIME File System Project Report, prepared by the Advanced Research Section, Research and Development Division, Oakland Police Department, Oakland, California 94607, December, 1972.

[2] Knuth, Donald E., The Art of Computer Programming v.3., Sorting and Searching, Reading, Massachusetts (Addison-Wesley), 1973.

[3] Oakland Police Department "CRIME" File System Internal Maintenance Specification, prepared by Hewlett-Packard Co., Data Systems-Operations Division, Custom Products Department, Cupertino, California 95014, February, 1973.

MUG FILE PROJECT REPORT UHMUG - 15

The UHMFS Computer Software
Part 2

by

George W. Batten, Jr.
Asif A. Karachiwala
Ha Hoang Nguyen

University of Houston
Houston, Texas

October 1, 1977

This project was supported by grant number 76-NI-99-012 awarded by the Law Enforcement Assistance Administration, U. S. Department of Justice, under the Omnibus Crime Control and Safe Streets Act of 1968, as amended. Points of view or opinions stated in this document are those of the authors and do not necessarily represent official position of the U. S. Department of Justice.

Table of Contents

Using the University of Houston Mug File System (UHMFS) Computer Software

Introduction

This is intended as a guide to the use of The University of Houston Mug File System (UHMFS). Although it is complete, there has been no effort to make it a user's manual for persons who are not already somewhat familiar with data processing systems. It is likely that any law enforcement agency adopting the present form of UHMFS would make some changes that would affect the user. Such changes should be reflected in a user's manual for the particular installation. If, as might be anticipated, future development of UHMFS results in a system which has general applicability, a corresponding manual would be appropriate. It is assumed that the reader is familiar with the Hewlett-Packard HP2100 DOS-III operating system, under which the present form of UHMFS runs.

The UHMFS programs are under control of the operator at the system console where the following functions can be initiated.

- Initialization of a new data base.
- Data base editing.
- Terminal activation to respond to the information-retrieval requests by investigative personnel.
- Listing specific records.
- Listing of an entire data base.

Other operations, such as preparing backup copies of the data base, are performed from the system console. However, these are considered as standard computer operating system operations and do not require special applications programs.

The Data Base - A User's Viewpoint

The data base of UHMFS is divided into two main parts, the Subject File (SF) and the Vehicle File (VF). Each of these parts comprises a number of records with information on crimes.

Each record of the Subject File contains all of the known (to the computer system) information on a single known-offender. Thus, each record contains typical physical descriptors (sex, height, weight, hair color, etc.), fingerprint codes, addresses (locations) of mugshots in a mugfile, and measurements of facial characteristics taken from the mugshots. The latter is a unique feature of UHMFS; it permits sorting of suspects on the basis of facial characteristics.

The records of VF contain information on vehicles which have been encountered in normal police activity; this information is gathered from such sources as vehicle citations and field reports. Since the information is gathered in a relatively small locality, it contains important clues to the solution of crimes committed in that area, and it can be very useful if a witness has seen a vehicle that is connected in some way with a crime.

In UHMFS, the files SF and VF are used independently of each other, but their use involves common principles of data-base management and information retrieval. Briefly, there must be some method of adding new information to these files, some method of purging out-of-date information from them, and some method of finding all those records which relate in some way to the clues for a particular crime. The adding of new information and the purging of old information are referred to collectively as "data-base management," and the recovery of files containing information based on certain key data is referred to as "information retrieval." In UHMFS, data-base management is handled by the updating subsystem comprising programs UPDAT, MERGE, RBLD, and EDIT (with input operations available under program QUERY). All information retrieval operations are done by program QUERY. Details of the use of these programs are given in the following sections.

The data-base files are organized to provide fast retrieval based on certain search parameters which are used frequently. This is accomplished by maintaining an indexing system (the key and reference files) for the master files. It is not necessary for the user to understand the data-base structure in detail. It is sufficient for him to know that mass-storrage space must be available for all these files. In UHMFS as it is implemented on the HP2100 computer with the DOS-III operating system, all files of the data base are maintained in three EFMP packs as follows:

Pack PN001	SF Master File,
Pack PN002	SF Key and Reference Files,
Pack PN003	VF Master, Key, and Reference Files.

The user needs to know of the presence of these packs only when initializating the system (usually done only once per data base) and when backing up the data base. These operations should be done only by the system manager.

The Updating Subsystem

The updating subsystem operates in four distinct phases: the input phase, the merging phase, the rebuilding phase, and the editing phase. The first three of these must occur in the order given; the editing phase can be done at any time. From a user's viewpoint, the merging and the rebuilding phases appear as a single phase, which we will call the merging/rebuilding phase.

The system operator starts the input phase by initiating operation of program UPDAT via the directive

```
:PR,UPDAT
```

typed on the system console if program QUERY is not operating, or by typing the command UPDAT on the system console if QUERY is operating (this cannot be done from a remote terminal). The system reads the update specification records (described in Appendix A) via the system card reader, checks them for detectable errors, reports any such errors on the system list device, and stores all correct records on the system mass storage device (disk). At this point checking is done on a single card basis; no check is made to determine if cards or fields of information are duplicated (as might occur, for example, if there are multiple corrections to a particular record of a master file). When the system reads the update specification end card, it notifies the operator and terminates the input phase. No changes have yet been made to the system data base, so QUERY operations can be done as if the input phase had not been started.

For the merging/rebuilding phase, QUERY operation must be terminated. This phase takes a considerable amount of time (several hours for a full data base), so it should be scheduled for a time when QUERY operations are not important. The system operator starts it by initiating operation of program MERGE via the directive

```
:PR,MERGE
```

typed on the system console. The system notifies the operator when this phase is completed.

The editing phase is described in a later section.

The Information Retrieval Subsystem - Initiation of Operation

All information retrieval is done via program QUERY, which the operator starts by typing the directive

```
:PR,QUERY
```

on the system console. The system begins executing this program which issues a sequence of requests to the operator.

These requests, and those to QUERY system users, appear below. Each will be followed on the succeeding line by a character in column one. The character will indicate to the system operator whether a code or numerical value is to be entered on the keyboard. This character which is called a prompt character, is always typed by the system. The parameter is entered on the keyboard immediately following the prompt character. All parameters are terminated by a carriage-return on the keyboard.

The two types of prompt characters are the up-arrow (↑) and the cross-hatch (#). The up-arrow (↑) character indicates that the parameter to be entered is in alpha-numeric code. The cross-hatch (#) character indicates that the parameter to be entered is a number (signed or unsigned) or a series of numbers separated by commas.

At any time throughout operation at a teleprinter terminal the user can enter the command "HELP," to which system will respond with instructions as to what options are available at this point. If the user should enter an illegal code, the system will print four question marks such as "????." In this case, the user should enter the correct response or ask for "HELP." The system will always respond with a prompt character when a new entry is required.

```
ENTER THE CURRENT YEAR 19##
```

The operator enters the last two digits of the current year. A response of less than 72 or greater than 99 will cause the request to be repeated.

```
ENTER THE USER STATUS CODE:
4 DIGITS
# — — — —
```

The operator enters a decimal number with an absolute value of less than 10,000; the number may be signed. Any other type of entry will cause the request to be repeated. This code is used to identify the data base disc cartridges to ensure that different date disc cartridges are not mixed together.

```
USER STATUS CODE DOES NOT MATCH!
QUERY TERMINATED
@
```

If the User Status Code does not match the identifying codes on all three (3) disc cartridges containing the SF and VF files, then these messages are typed in response to the status code entry above. The program terminates after this message.

```
IDENTIFY THE ACTIVE TERMINALS
BY LUN (UP TO 6)
#— —, — —, — —, — —, — —, — —
```

The operator enters up to six logical unit numbers (LUN), each separated by a comma, to indicate the active teleprinter terminals to be used. An entry that is negative, less than

	7, greater than 99 or otherwise illegal causes the request to be repeated. A check is made of each LUN to make sure that it corresponds to a teleprinter-like device.
QUERY SYSTEM IS NOW OPERATIONAL	This message is typed when all entries have been made and the query terminals are ready.

The Information Retrieval Subsystem - QUERY Terminal Operations

Query terminals in the UHMFS File System comprise a teleprinter and an image retrieval unit (microfiche). Each of these terminals can be utilized for any one of three query applications: mug shot display, fingerprint display, or vehicle search. The mug shot and fingerprint applications use the teleprinter in conjunction with the image retrieval unit. The vehicle search application utilizes the teleprinter alone. In each of these applications, the user enters commands and query codes through the teleprinter and receives back "hit" information through the image retrieval unit and/or the teleprinter.

To initiate terminal operation, the user enters a query type code on the terminal teleprinter keyboard. The query type codes that can be entered are

1) MS for the mug shot display,

2) FP for the fingerprint display, and

3) VF for the Vehicle File

(these must be followed by a carriage return, of course). This alerts the system which prepares for operations on the appropriate file and responds with "READY" and the prompt character "↑".

When the terminal is ready, the user should enter the appropriate function command. There are four basic functions for the mug shot and fingerprint searches, and two basic functions for use in the vehicle search. In addition to these functions, there is a command (END) to terminate the current MS, FP or VF operation. The commands follow.

1) ID — for purpose of obtaining the current image addresses of any given subject in the SF file (not available in VF mode).

2) SEARCH — for entering the query codes for a search of the Subject or Vehicle Files.

3) PRINT — allows the user to have a summary of the current "hits" listed on the teleprinter for inspection or documentation purposes.

4) DISPLAY — allows the user to display either the mug shot or fingerprint photographs on the respective image retrieval unit for the current list of hits (not available in VF mode).

5) END — terminate current MS, FP or VF operation.

Only the first two characters of each command word need be typed. Detailed descriptions of the functional commands are in the sections that follow.

The ID Command

This command is available only in the mug shot and fingerprint file search modes because these are the only modes associated with images. With this command the user can obtain the microfilm addresses for a particular subject so that he can inspect the subject's mug shot(s) or fingerprints. The subject is selected by his police identification number. The command and number entered as

```
↑ID
#nnnnnnn
```

where nnnnnnn is the subject's identification number. The UHMFS response will be of the form

$$\text{aaaaa} \qquad \underbrace{\text{aaaaa} \qquad \text{aaaaa} \qquad \text{aaaaa}}_{\text{(Used if necessary for mugshots only.)}}$$

where aaaaa is the address of the image in the image retrieval unit. The prompt character "#" will be repeated if the ID number entered is illegal. If a legal number is entered but there is no subject with this ID number, in the system, then the message "NON EXISTENT ID" will be printed followed by the prompt character "↑" on the next line.

<u>The SEARCH Command</u>

This command which is the major query command in the system, if for the purpose of informing the system that

1) a totally new search is being made so any list of "hits" from a previous query operation at that terminal should be cleared, and
2) the user wishes to enter a sequence of query codes to be used as parameters in a new search.

The query codes are entered on the teleprinter keyboard from the Subject File Input (SFI) Form (Appendix B) following the prompt character. Usually, only one query code per line is allowed. Exceptions to this rule are for those query codes (as specified on the SFI form) that can be used in combinations as logical "or" functions. A maximum of two responses can be combined for most query codes. These combinations of query codes are entered on one line separated by a comma. There can be no intervening blanks in the entries. Query parameters which are numeric should be entered in the manner specified on the SFI form. Examples of query code entries are given below with their meanings.

↑A1 The subject is male

↑B2,B3 The subject is either black or brown

↑C1 The subject is between 20 and 25 years old.
#20,25

↑J2,J4,J8 The subject is partly bald or thinning, and he has medium length hair, which is curly, wavy or kinky.

Fingerprint codes, which are in the single fingerprint classification scheme used by the Oakland Police Department (California), (see Single Fingerprint Classification Scheme [1]), are slightly different in format from the other query codes. The user can specify the fingerprint codes while searching the Subject File for mug shot or fingerprint displays. Up to three fingers can be described for a search operation. Fingerprints are specified in terms of both finger numbers and fingerprint codes. All specifications are in numerical terms. The code "?" may be used to replace the finger number and/or up to two digits in the three digit fingerprint code when these parameters are unknown. Only one finger number can be specified as unknown. When a finger number is unknown, then an automatic search is made on all fingers (0-9) in order. Examples of fingerprint query entries follow:

↑W1_ The subject is known to have code 123 on finger number 0.

↑W1_ The subject is known to have some finger with code 111.
#?,111

↑W1 #1,112;?,12?;4,3??	The subject is known to have code 112 on finger number 1; code 12? on some finger; and code 3?? on finger number 4.

If either a query code or fingerprint entry is entered incorrectly for any reason, then the message "ENTRY ERROR" is typed, the entire line is ignored, and the prompt character is repeated.

When the user finishes entering search parameters, he types the command DONE. The system searches the data base for records matching the search parameters. When the search is complete, the system types the message

```
THERE ARE nnn HITS
```

where nnn is the number of records matching the search parameters.

If the operational mode is MS, the system asks

```
WANT TO USE THE LOOK-ALIKE ALGORITHM? (YES OR NO)
```

to which the user must respond appropriately. If he responds with YES, the system checks to see if facial measurements have already been supplied (i.e., via code Z1 when the search parameters were being entered). If so, it skips the next step and sorts the hit list as described below; otherwise it requests them by typing

```
ENTER FACIAL DATA TYPE (PHOTO, SKETCH, OR COMPOSITE)
```

to which the user must give an appropriate answer. The computer responds

```
ENTER FACIAL MEASUREMENTS, ONE AT A TIME AS ASKED FOR
#1:
```

thereby requesting facial data. After the user has entered all of the measurements as required, the computer sorts the hit list into order so that the "best look-alike" is first, then it types

```
THE 'HITS' ARE NOW ORDERED ACCORDING TO THEIR FACIAL DISTANCES
```

and the user can enter another command.

If the operational mode is not MS, the operations in the paragraph above are skipped. The same is true if the user's response to the question of that paragraph is "NO."

If the user considers the number of hits to be too large, his only recourse other than terminating the search at this point is to reduce the size of the list by specifying more search parameters; this should be done without giving the SEARCH command. Otherwise, the hit list can be printed (via the PRINT command) or images corresponding to records in the hit list can be displayed (via the DISPLAY command).

The PRINT Command

This command causes a summary of the current "hit" list to be printed on the teleprinter. The information is printed in one of two formats depending on the mode of the terminal: (MS or FP) or Vehicle File Search (VF). Both formats have one line per hit. For the Subject File, the format is the ID number followed by the subject name: for example,

```
123456  THWART, AFT.
```

For the Vehicle File, the format is the name, the sex, the race, the date of birth, and the date of citation in that order; for example:

```
STRAKE, BILGE  M W 10/4/46   16/2/76
```

The DISPLAY Command

This command causes addresses corresponding to records in the current hit list to be sent to the image retrieval unit. Details of the use of this command are omitted from this report since they are dependent on the particular implementation of the system (some image retrieval units have an associated buffer to hold address lists).

The END Command

This command terminates the current mode (MS, FP, or, VF) of operation so that another mode can be entered.

EDIT Operations

The editing phase operates only when program QUERY is not active. The system operator initiates operation of this phase by typing

```
:PR,EDIT
```

on the system console. The system responds as follows.

```
ENTER EDIT FILE TYPE: "SF" or "VF".
↑ — —
```

This response indicates to UHMFS which of the two major files is to be updated; the "Subject file" (SF) or the "Vehicle file" (VF). Any other responses will cause the question to be repeated.

```
ENTER "OLD" USER STATUS
CODE:  4 DIGITS
#— — — —
```

The operator enters a decimal number with an absolute value of less than 10,000; the number may be signed. Any other entry causes the request to be repeated. This code is used to identify the "old" data base disc cartridges to ensure that different-date disc cartridges are not mixed together.

```
USER STATUS CODE DOES NOT MATCH!
UPDATE TERMINATED.
```

If the User Status Code does not match the identifying codes on all three disc cartridges containing the SF and VF files, then these messages are typed in response to the "old" status code entry above. The program terminates after these messages.

Once proper responses have been given to the requests above, the system responds with the up-arrow (↑) prompt character. This is a request for one of the following editing-phase commands.

1) EDIT — to change certain fields of one record; fields which can be changed are marked with an asterisk (*) in Appendix A.

2) DUMP — to list the entire contents of the file (SF or VF) being edited onto the system list device.

3) PURGE — to delete old references in the file by citation date (VF) or year of arrest (SF).

4) DATE — to list all file entries in the Vehicle File with a particular citation date (not available in SF mode).

5) ID — to list a particular subject entry by the Oakland Police Department (OPD) identification number (not available in VF mode).

6) DELETE — to delete a specific record from the file (SF or BF) being edited.

7) END — to terminate the editing phase.

In each case the system responds to the user in a self-explanatory manner. The user must give appropriate answers to questions asked.

References

[1] Single Fingerprint Classification Scheme, a report prepared by The Criminalistics Section, Oakland Police Department, Oakland, California 95014, March, 1973.

Appendix A

Update Specification Record Formats

Formats of records for input to the UHMFS updating subsystem are given in the following table. Note the special format of cards for deleting records (--DELETE-- cards) and the end-of-input cards (--END-- cards). Figures A-1, A-2, and A-3 are forms for data input and query. Figure A-4 shows the suggested input card layout.

Table A-1

CARD FORMATS

CARD #1: SUBJECT FILE

COL.#	DESCRIPTION	WIDTH	CONTENTS
1	Card #	1	1
2	ID Prefix	1	Indicates type of Identification Number
3-9	ID No.	7	Identification Number
10-33	Name	24	Alpha/Numeric Characters for Last Name, First Name, Middle Initial.
34	Sex	1	1= Male 2= Female
35	Race	1	1= White 2= Black 3= Brown 4= Yellow 5= Undetermined
36-37	Year of Birth	2	2-digit number from 00 to 99*
38-39	"Beat" Number	2	2-digit number from 00 to 99*
40-41	Type of Crime #1*	2	01= Commercial Burglary 02= Residential Burglary 03= Locked Auto Burglary 04= Possession of Stolen Property 05= Ignored 06= Armed Robbery 07= Strongarm and Purse Snatch 08= Ignored 09= Auto Theft 10= Grand Theft 11= Check and Credit Card Theft 12= Felony Assault 13= Murder 14= Ignored 15= Rape 16= Indecent Exposure 17= Child Molesting 18= Other-Sex 19= Ignored 20= Narcotics and Drugs 21= Fraud and Bunco 22= Arson 23= Miscellaneous Felonies 24= Operates in teams (major crimes only, such as robbery and burglary). 25= Associated with shoplifting ring 26= Associated with check/credit card ring 27= Associated with group advocating or practicing violence

*Leading zeros are required.

CARD #1: SUBJECT FILE (CONTINUED)

COL.#	DESCRIPTION	WIDTH	CONTENTS
40-41	Type of Crime #1* (Continued)	2	28= Associated with extortion/ loan shark ring 29= Associated with consumer fraud/bunco ring 30= Associated with organized prostitution 31= Associated with organized narcotics 32= Associated with organized gambling 33= Associated with auto stripping ring
42-43	Type of Crime #2*	2	Same as Type of Crime #1 plus Blanks=None
44-45	Height (in inches).	2	2 digit no. from 01 to 99*
46-47	Weight (in pounds)	3	3 digit no. from 001 to 999*
48	Hair Color	1	1= Blond 2= Brown 3= Black 4= Red 5= White or Grey
49	Hair Style #1	1	1= Bald 2= Partly bald or thinning 3= Close cut or short 4= Medium Length 5= Long 6= Afro American-Natural Style 7= Afro American-Processed Style 8= Curly, Wavy, Kinky 9= Straight
50	Hair Style #2	1	Same as Hair Style #1 plus Blanks=None
51	Hair Style #3	1	Same as Hair Style #2 plus Blanks=None
52	Eye Color	1	1= Brown or Black 2= Blue, Grey, Green or Hazel
53	Eye Defects	1	Blank=None of Below 1= Either eye blind, missing or artificial 2= Wears glasses(prescription)
54	Ears	1	Blank=None of Below 1= Cauliflower 2= Partial or missing 3= Excessively protruding 4= Male with earring(s)
55	Lips	1	Blank=None of Below 1= Harelip 2= Unusually large 3= Other permanent deformity

*Leading zeros are required.

CARD #1: SUBJECT FILE (CONTINUED)

COL.#	DESCRIPTION	WIDTH	CONTENTS
56	Complexion	1	Blank=None of Below 1= Light, fair 2= Dark, Black 3= Freckled or splotchy 4= Pockmarked
57	Tattoo Marks	1	Blank=None of Below 1= Arms 2= Hands or fingers 3= Face and Neck 4= Other location or combinations of above
58	Facial Hair	1	Blank=None 1= Yes
59	Teeth	1	Blank=None of Below 1= Irregular or protruding 2= Metal fillings visible 3= Visible cecay or stains 4= False, chipped or missing teeth
60	Amputations and Deformities	1	Blank=None of Below 1= Arms 2= Hands or Fingers 3= Legs or Feet 4= Other or combination of 1 thru 3
61	Visible scars, moles, birthmarks, or needle mark tracks	1	Blank=None of Below 1= Arms 2= Hands or Fingers 3= Face, Head or Neck 4= Other or combination of 1 thru 3
62	Speech #1	1	Blank=None of Below 1= Foreign Accent 2= Noticeable regional accent 3= Lisp 4= Stutter
63	Speech #2	1	Same as Speech #1
64	Peculiarities	1	Blank=None of Below 1= Limp 2= Effeminate Male or Masculine Female 3= Wears clothing of opposite sex or Impersonator 4= Twitch of eye(s), face or other
65-74	Nickname	10	Alpha/Numeric Characters for Nickname
75-76	Year of arrest	2	2-digit no. from 00 to 99*

*Leading zeros are required.

CARD #2: SUBJECT FILE FORMAT

COL. #	DESCRIPTION	WIDTH	CONTENTS
1	Card #	1	2
2	ID Prefix	1	Same as for Card #1
3-9	ID	7	Same as for Card #1
10-14	Fingerprint Address	5	3-digit Fiche No., Row (A-G),Column (A-N)*
15-17	Fingerprint #0 Class.	3	3 digit numbers from 0 thru 9 each*
18-20	Fingerprint #1 Class.	3	Same as Fingerprint #0 Classification*
21-23	Fingerprint #2 Class.	3	Same as Fingerprint #0 Classification*
24-26	Fingerprint #3 Class.	3	Same as Fingerprint #0 Classification*
27-29	Fingerprint #4 Class.	3	Same as Fingerprint #0 Classification*
30-32	Fingerprint #5 Class.	3	Same as Fingerprint #0 Classification*
33-35	Fingerprint #6 Class.	3	Same as Fingerprint #0 Classification*
36-38	Fingerprint #7 Class.	3	Same as Fingerprint #0 Classification*
39-41	Fingerprint #8 Class.	3	Same as Fingerprint #0 Classification*
42-44	Fingerprint #9 Class.	3	Same as Fingerprint #0 Classification*
45-49	Mug Shot Address #1	5	3 digit Fiche No., Row (A-G), Column (A-N)*
50-54	Mug Shot Address #2	5	Same as Mug Shot Address #1 or Blank*
55-59	Mug Shot Address #3	5	Same as Mug Shot Address #1 or Blank*
60-64	Mug Shot Address #4	5	Same as Mug Shot Address #1 or Blank*

*Leading zeros are required.

CARD #3: MUG SHOT MEASUREMENTS

COL.#	DESCRIPTION	WIDTH	CONTENTS
1	Card #	1	3
2	ID Prefix	1	Same as for Card #1
3-9	ID	7	Same as for Card #1
10-12	Measurement #1	3	3 digit number-measurement #1*
13-15	Measurement #2	3	3 digit number-measurement #2*
16-18	Measurement #3	3	3 digit number-measurement #3*
19-21	Measurement #4	3	3 digit number-measurement #4*
22-24	Measurement #5	3	3 digit number-measurement #5*
25-27	Measurement #6	3	3 digit number-measurement #6*
28-30	Measurement #7	3	3 digit number-measurement #7*
31-33	Measurement #8	3	3 digit number-measurement #8*
34-36	Measurement #9	3	3 digit number-measurement #9*
37-39	Measurement #10	3	3 digit number-measurement #10*

*Leading zeros are required.

CARD #4: VEHICLE FILE

COL.#	DESCRIPTION	WIDTH	CONTENTS
1	Card#	1	4
2-19	Name	18	Alpha/Numeric Characters for Last Name, First Name, Middle Initial.

20	Sex	1	M= Male
			F= Female

21	Race	1	W= White
			N= Negro
			M= Mexican (Latin American)
			I= Indian, American
			C= Chinese
			J= Japanese
			O= All Others

22-27	Date of Birth	6	2-digit no. for day (01-31)* 2-digit no. for month (01-12)* 2-digit no. for year (00-99)*

28	Field Contact	1	Blank=No
			F= Yes

29-34	Date of Citation	6	2-digit no. for day (01-31)* 2-digit no. for month (01-12)* 2-digit no. for year (72-99)*
35-36	Year of Vehicle	2	2-digit no. (00-99)*

37-40	Make	4	ALFA= Alpha Romeo
			ALPI= Alpine
			AMER= American Motors
			ASTO= Aston-Martin
			AUDI= Audi
			AUST= Austin
			AUHE= Austin Healy
			BENT= Bentley
			BMC = BMC
			BMW = BMW
			BORG= Borgward
			BUIC= Buick
			CADE= Cadillac
			CAP = Capri (Import)
			CHEV= Chevrolet
			CHRY= Chrysler
			DATS= Datsun
			DESO= DeSoto
			DODG= Dodge
			EDSE= Edsel
			ENGF= English Ford (British)
			FERR= Ferrari
			FIAT= Fiat
			FIAA= Fiat-Abarth
			FORD= Ford
			HILL= Hillman
			HOND= Honda
			INTL= International (Harvester)
			JAGU= Jaguar
			JEP = Jeep
			KARG= Karmann-Ghia
			LINC= Lincoln
			LOTU= Lotus
			MAZD= Mazda

*Leading zeros are required.

CARD #4: VEHICLE FILE FORMAT (CONTINUED)

COL.#	DESCRIPTION	WIDTH	CONTENTS
37-40	Make(Continued)	4	MERZ= Mercedes-Benz
			MERC= Mercury
			MG = MG
			NASH= Nash
			OLDS= Oldsmobile
			OPEL= Opel
			PACK= Packard
			PEUG= Peugeot
			PLY = Plymouth
			PONT= Pontiac
			PORS= Porsche
			RAMB= Rambler
			RENA= Renault
			ROL = Rolls-Royce
			SAA = Saab
			SHEB= Shelby American
			SPEC= Special Vehicle
			STU = Studebaker
			SUBA= Subaru
			SUNB= Sunbeam
			SUZI= Suzuki
			TOYT= Toyota
			TRIU= Triumph
			VOLK= Volkswagen
			VOLV= Volvo
			WILL= Willys-Overland
			BSA = BSA
			HD = Harley-Davidson
			HONM= Honda Motorcycle
			KAWK= Kawasaki
			SUZI= Suzuki
			TRIM= Triumph Motorcycle
			YAMA= Yamaha
41-43	Model	3	Spaces=None of Below
			AMB= Ambassador
			AMX= AMX
			GRE= Gremlin
			HOR= Hornet
			JAV= Javelin
			RA = Rambler American
			RIV= Riviera
			SKY= Skylark
			ELD= El Dorado (FLE)
			CAM= Camaro
			CAP= Caprice
			ELL= Chevelle
			CH2= Chevy II
			CVR= Corvair
			CVT= Corvette
			ELC= El Camino
			IMP= Impala
			MAL= Malibu (ELL)
			NOV= Nova (CH2)
			VEG= Bega
			IMP= Imperial
			CHL= Challenger
			CHA= Charger
			COR= Coronet
			DAR= Dart

CARD #4: VEHICLE FILE FORMAT (CONTINUED)

COL.#	DESCRIPTION	WIDTH	CONTENTS
41-43	Model (Continued)	3	COB= Cobra
			FAI= Fairlane
			FAL= Falcon
			GAL= Galaxie
			LTD= LTD
			MAV= Maverick
			MUS= Mustang
			PIN= Pinto
			RAN= Ranchero
			THU= Thunderbird
			TOR= Torino (FAI)
			CAP= Capri
			CON= Continental
			COM= Comet
			COU= Cougar
			MET= Metropolitan
			RAM= Rambler
			CUT= Cutlass (F-85)
			BAR= Barracuda
			BEL= Belvedere
			FUR= Fury
			GTX= GTX
			RRU= Road Runner
			SAT= Satellite
			VAL= Valiant
			BON= Bonneville
			FRD= Firebird
			GRA= Grand Prix
			GTO= GTO
			LEM= LeMans
			TEM= Tempest
			AME= American
			DUN= Dune Buggy
44-45	Body Type	2	2D = 2-Door
			4D = 4-Door
			SW = Station Wagon
			CV = Convertible
			PU = Pickup
			SP = Sport Car
			VN = Van
			BS = Bus
			HS = Hearse
			MC = Motorcycle
46-48	Color #1	3	BGE= Beige
			BLK= Black
			BLU= Blue
			LBL= Light Blue
			DBL= Dark Blue
			BRZ= Bronze
			BRO= Brown
			CRM= Cream
			GLD= Gold
			GRN= Green

CARD #4: VEHICLE FILE FORMAT (CONTINUED)

COL.#	DESCRIPTION	WIDTH	CONTENTS

46-48	Color #1 (cont.)	3	LGR= Light Green
			DGR= Dark Green
			GRY= Grey
			LAV= Lavender
			ONG= Orange
			MAR= Maroon
			RED= Red
			SIL= Silver
			TAN= Tan
			TRQ= Turquoise
			WHI= White
			YEL= Yellow
			PNK= Pink
			PLE= Purple

49-51	Color#2	3	Same as Color #1 or Blank if not 2-tone

52	State	1	C= California
			O= Other or Out-of-State

53-58	License Number	6	Alpha/Numeric Characters (Left justified with all blanks (spaces) squeezed out)

RECORD DELETION CARD FORMAT

This card has the format of Card 1 (Subject File) or Card 2 (Vehicle File) except for columns 65 - 74 which must contain --DELETE--. The corresponding record of SF or VF is deleted. The ID number must be given for SF; the name must be given for VF. Fields which are not blank are tested for agreement with the record of the master file; if there is any disagreement, the record is not deleted.

UPDATE END CARD FORMAT

Col.#	Description	Width	Contents
1-7	Card Identification	7	--END--

SUBJECT FILE INPUT (SFI) FORM

ID NUMBER (ENTER CII OR OPD NUMBER)

_ _ _ _ _ _ _

DATE ______________

(NEW) ________
(UPDATE) ________ } CHECK ONE

NAME (ENTER UP TO 24 CHARACTERS): _
LAST, FIRST MIDDLE

SEX

(CIRCLE ONE)
1 Male
2 Female

RACE

(CIRCLE ONE)
1 White
2 Black
3 Brown
4 Yellow
5 Undetermined

AGE

(ENTER YEAR OF BIRTH)

19 __ __

RESIDENCE CODE

(ENTER TWO-DIGIT "BEAT" NUMBER)

(__ __)

TYPE OF CRIME

(CIRCLE ONE OR TWO)
1 Commercial burglary
2 Residential burglary
3 Locked auto burglary
4 Possession of stolen property
NOT USED FOR UPDATING
6 Armed robbery
7 Strongarm and purse snatch
NOT USED FOR UPDATING
9 Auto theft
10 Grand theft
11 Check and credit card
12 Felony assault
13 Murder
NOT USED FOR UPDATING
15 Rape
16 Indecent exposure
17 Child molesting
18 Other-sex
NOT USED FOR UPDATING
20 Narcotics and drugs
21 Fraud and bunco
22 Arson
23 Multiple felonies
24 Operates in teams (major crimes only, such as robbery and burglary)
25 Associated with shoplifting ring
26 Associated with check/credit card ring
27 Associated with group advocating or practicing violence
28 Associated with extortion/loan shark ring
29 Associated with consumer fraud/bunco ring
30 Associated with organized prostitution
31 Associated with organized narcotics
32 Associated with organized gambling
33 Associated with auto stripping ring

HEIGHT

(ENTER HEIGHT)

(__ __ INCHES)

WEIGHT

(ENTER WEIGHT)

(__ __ __ POUNDS)

HAIR COLOR Δ

(CIRCLE ONE)
1 Blond
2 Brown
3 Black
4 Red
5 White or grey

HAIR STYLE Δ

(CIRCLE ONE, TWO, OR THREE)
1 Bald
2 Partly bald or thinning
3 Close cut or short
4 Medium length
5 Long (below ears)
6 Afro american-natural style
7 Afro american-processed style
8 Curly, wavy, or kinky
9 Straight

EYE COLOR Δ

(CIRCLE ONE)
1 Brown or black
2 Blue, grey, green, or hazel

EYE DEFECTS Δ

(CIRCLE ONE)
1 Either eye blind, missing, or artificial
2 Wears glasses (prescription)
3 1 or 2

EARS Δ

(CIRCLE ONE)
1 Cauliflower
2 Partial or missing
3 Excessively protruding
4 Male with earring(s)

LIPS Δ

(CIRCLE ONE)
1 Hairlip
2 Unusually large
3 Other permanent deformity

COMPLEXION Δ

(CIRCLE ONE)
1 Light, fair
2 Dark, black
3 Freckled or splotchy
4 Pockmarked

TATTOO MARKS Δ

(CIRCLE ONE)
1 Arms
2 Hands or fingers
3 Face and neck
4 Other location or combinations of the above

FACIAL HAIR Δ

(CIRCLE)
1 Yes

TEETH Δ

(CIRCLE ONE)
1 Irregular or protruding
2 Metal fillings visible
3 Visible decay or stains
4 False, chipped, or missing teeth

AMPUTATIONS & DEFORMITIES Δ

(CIRCLE ONE)
1 Arms
2 Hands or fingers
3 Legs or feet
4 Other or combinations of the above

VISIBLE SCARS, MOLES, BIRTHMARKS, OR NEEDLE TRACKS Δ

(CIRCLE ONE)
1 Arms
2 Hands or fingers
3 Face, head, or neck
4 Other location or combinations of the above

SPEECH Δ

(CIRCLE ONE)
1 Foriegn accent
2 Noticeable regional accent
3 Lisp
4 Stutter

PECULIARITIES Δ

(CIRCLE ONE)
1 Limp
2 Effeminate male or masculine female
3 Wears clothing of opposite sex (impersona
4 Twitch of eye(s), face, or other

NICKNAME

(ENTER UP TO 10 CHARACTERS, IF NONE, ENTER FIRST NAME)

_ _ _ _ _ _ _ _ _ _

Δ NOTE:
(FOR CATEGORIES H THRU W RESPOND WHERE THE APPROPRIATE CONDITION EXISTS; ALL OTHER CATEGORIES REQUIRE A RESPONSE.)

MUG SHOT ADDRESSES

________ ________ ________ ________

FICHE ADDRESS ________

FINGERPRINT CODES

(ENTER 10 THREE-DIGIT CODES)

(_ _ _ | _ _ _ | _ _ _ | _ _ _ | _ _ _ | _ _ _ | _ _ _ | _ _ _ | _ _ _ | _ _ _)
0 1 2 3 4 5 6 7 8 9

SUBJECT FILE QUERY FORM (SFQ)

ID NUMBER

_ _ _ _ _ _ _

DATE ________

NAME _
LAST, FIRST MIDDLE

SEX

[CIRCLE ONE]
A1 Male
A2 Female

RACE

[CIRCLE ONE OR TWO]
B1 White
B2 Black
B3 Brown
B4 Yellow
B5 Undetermined

AGE

[ENTER MINIMUM & MAXIMUM AGE LIMITS]
C1

[# _ _,_ _ YEARS OF AGE]

RESIDENCE CODE

[ENTER UP TO 3 ONE-DIGIT DISTRICT NOS.]
D1

[# _,_,_]

TYPE OF CRIME

[CIRCLE ONE OR TWO]
E1 Commercial burglary
E2 Residential burglary
E3 Locked auto burglary
E4 Possession of stolen property
E5 ✱ E1, E2, E3, or E4
E6 Armed robbery
E7 Strongarm and purse snatch
E8 ✱ E6 or E7
E9 Auto theft
E10 Grand theft
E11 Check and credit card
E12 Felony assault
E13 Murder
E14 ✱ E12 or E13
E15 Rape
E16 Indecent exposure
E17 Child molesting
E18 Other sex
E19 ✱ E15, E16, E17, or E18
E20 Narcotics and drugs
E21 Fraud and bunco
E22 Arson
E23 Multiple felonies
E24 Operates in teams (major crimes only, such as robbery and burglary)
E25 Associated with shoplifting ring
E26 Associated with check/credit card ring
E27 Associated with group advocating or practicing violence
E28 Associated with extortion/loan shark ring
E29 Associated with consumer fraud/bunco ring
E30 Associated with organized prostitution
E31 Associated with organized narcotics
E32 Associated with organized gambling
E33 Associated with auto stripping ring

HEIGHT

[ENTER MINIMUM AND MAXIMUM HEIGHT LIMITS]
F1

[# _ _,_ _ INCHES]

WEIGHT

[ENTER MINIMUM AND MAXIMUM WEIGHT LIMITS]
G1

[# _ _ _,_ _ _ POUNDS]

HAIR COLOR Δ

[CIRCLE ONE OR TWO]
H1 Blond
H2 Brown
H3 Black
H4 Red
H5 White or grey

HAIR STYLE Δ

[CIRCLE ONE, TWO, OR THREE]
J1 Bald
J2 Partly bald or thinning
J3 Close cut or short
J4 Medium length
J5 Long (below ears)
J6 Afro american-natural style
J7 Afro american-processed style
J8 Curly, wavy, or kinky
J9 Straight

EYE COLOR Δ

[CIRCLE ONE]
K1 Brown or black
K2 Blue, grey, green, or hazel

EYE DEFECTS Δ

[CIRCLE ONE]
L1 Either eye blind, missing, or artificial
L2 Wears glasses (prescription)
L3 L1 or L2

EARS Δ

[CIRCLE ONE]
M1 Cauliflower
M2 Partial or missing
M3 Excessively protruding
M4 Male with earring(s)

LIPS Δ

[CIRCLE ONE]
N1 Hairlip
N2 Unusually large
N3 Other permanent deformity

COMPLEXION Δ

[CIRCLE ONE OR TWO]
P1 Light, fair
P2 Dark, black
P3 Freckled or splotchy
P4 Pockmarked

TATTOO MARKS Δ

[CIRCLE ONE]
Q1 Arms
Q2 Hands or fingers
Q3 Face and neck
Q4 Other location or combinations of the above

FACIAL HAIR Δ

[CIRCLE]
R1 Yes

TEETH Δ

[CIRCLE ONE OR TWO]
S1 Irregular or protruding
S2 Metal fillings visible
S3 Visible decay or stains
S4 False, chipped, or missing teeth

AMPUTATIONS & DEFORMITIES Δ

[CIRCLE ONE]
T1 Arms
T2 Hands or fingers
T3 Legs or feet
T4 Other or combinations of the above

VISIBLE SCARS, MOLES, BIRTH-MARKS, OR NEEDLE TRACKS Δ

[CIRCLE ONE]
U1 Arms
U2 Hands or fingers
U3 Face, head, or neck
U4 Other location or combinations of the above

SPEECH Δ

[CIRCLE ONE OR TWO]
V1 Foriegn accent
V2 Noticeable regional accent
V3 Lisp
V4 Stutter

PECULIARITIES Δ

[CIRCLE ONE]
W1 Limp
W2 Effeminate male or masculine female
W3 Wears clothing of opposite sex (impersonator)
W4 Twitch of eye(s), face, or other

NICKNAME

[ENTER UP TO 10 CHARACTERS, IF NONE, ENTER FIRST NAME]
X1
= _ _ _ _ _ _ _ _ _ _

MUG SHOT ADDRESSES

____ ____ ____ ____

FINGERPRINT FICHE ADDRESS ________

FINGERPRINT CODES

[ENTER FINGER NUMBER AND THREE-DIGIT CODE, UP TO THREE TIMES; USE "?" FOR UNKNOWN CHARACTERS]
Y1

[# _,_ _ _ ; _,_ _ _ ; _,_ _ _]

Figure A-2

VEHICLE FILE INPUT AND QUERY FORM (VFI/VFQ)

CHECK HERE FOR QUERY ONLY □. DATE ____________

FIELD CONTACT

CIRCLE ONE

A1	Yes
A2	No

DATE OF CITATION

ENTER CITATION PERIOD AS "DAY, MONTH, & YEAR", 1 OR 2 SIX-DIGIT DATES

B1

#_ _ _ _ _ _,_ _ _ _ _ _

DDMMYY DDMMYY

YEAR OF VEHICLE

ENTER TWO-DIGIT YEAR NUMBER

C1

#19_ _

MAKE/MODEL

CIRCLE ONE MAKE AND A MODEL IF APPLICABLE

D1	Alpha Romeo
D2	Alpine
D3	American Motors
#0	No specific model
#1	Ambassador
#2	AMX
#3	Gremlin
#4	Hornet
#5	Javelin
#6	Rambler American (See D38-#2/D46-#1)
D4	Aston-Martin
D5	Audi
D6	Austin
D7	Austin Healey
D8	Bentley
D9	BMC
D10	BMW
D11	Borgward
D12	Buick
#0	No specific model
#1	Riviera
#2	Skylark
D13	Cadillac
#0	No specific model
#1	El Dorado (FLE)
D14	Capri (import)
D15	Chevrolet
#0	No specific model
#1	Camero
#2	Caprice
#3	Chevelle
#4	Chevy II
#5	Corvair
#6	Corvette
#7	El Camino
#8	Impala
#9	Malibu (ELL)
#10	Monte Carlo
#11	Nova (CH2)
#12	Vega
D16	Chrysler
#0	No specific model
#1	Imperial
D17	Datsun
D18	Desoto
D19	Dodge
#0	No specific model
#1	Challenger
#2	Charger
#3	Coronet
#4	Dart
Dune Buggy (See D51-#1)	

MAKE/MODEL -CONTINUED-

AUTOMOBILES -CONTINUED-

D20	Edsel
D21	English Ford (British)
D22	Ferrari
D23	Fiat
D24	Fiat-Aborth
D25	Ford
#0	No specific model
#1	Cobra
#2	Fairlane
#3	Falcon
#4	Galaxie
#5	LTD
#6	Maverick
#7	Mustang
#8	Pinto
#9	Ranchero
#10	Thunderbird
#11	Torino (FAI)
D26	Hillman
D27	Honda
D28	International (Harvester)
D29	Jaguar
D30	Jeep
D31	Karmann-Ghia
D32	Lincoln
#0	No specific model
#1	Capri
#2	Continental
D33	Lotus
D34	Mazda
D35	Mercedes-Benz
D36	Mercury
#0	No specific model
#1	Comet
#2	Couger
D37	MG
D38	Nash
#0	No specific model
#1	Metropolitan
#2	Rambler (See D3-#6/D46-#1)
D39	Oldsmobile
#0	No specific model
#1	Cutlass (F-85)
D40	Opel
D41	Packard
D42	Peugeot
D43	Plymouth
#0	No specific model
#1	Barracuda
#2	Belvedere
#3	Duster
#4	Fury
#5	GTX
#6	Road Runner
#7	Satellite
#8	Valiant
D44	Pontiac
#0	No specific model
#1	Bonneville
#2	Firebird
#3	Grand Prix
#4	GTO
#5	LeMans
#6	Tempest
D45	Porsche
D46	Rambler
#0	No specific model
#1	American (See D3-#6 D38-#2)
D47	Renault

MAKE/MODEL -CONTINUED-

AUTOMOBILES -CONTINUED-

D48	Rolls-Royce
D49	Saab
D50	Shelby American (See D25-#7)
D51	Special Vehicle
#0	No specific model
#1	Dune Buggy
D52	Studebaker
D53	Subaru
D54	Sunbeam
D55	Suzuki
D56	Toyota
D57	Triumph
D58	Volkswagon
D59	Volvo
D60	Willys-Overland

MOTORCYCLES

D61	BSA
D62	Harley Davidson
D63	Honda
D64	Kawasaki
D65	Suzuki
D66	Triumph
D67	Yamaha

BODY TYPE

CIRCLE ONE

E1	2-door
E2	4-door
E3	Station Wagon
E4	Convertible
E5	Pickup
E6	Sportscar
E7	Van
E8	Panel
E9	Bus
E10	Hearse

COLOR

CIRCLE ONE (SINGLE COLOR) OR TWO (2 TONE)

T/S*	B*	
F1	F1	Black
F2	F2	Brown, Beige, Bronze, or Tan
F3	F3	Red, Pink, or maroon
F4	F4	Orange
F5	F5	Yellow or Gold
F6	F6	Green or Turquoise
F7	F7	Blue
F8	F8	Purple or Lavender
F9	F9	Silver or Grey
F10	F10	White or Cream

*NOTE:
T/S=TOP OR SINGLE COLOR
B=BOTTOM COLOR

STATE

CIRCLE ONE

G1	California
G2	Other

LICENSE NUMBER

ENTER UP TO 6 CHARACTERS LEFT JUSTIFIED WITH ALL BLANKS (SPACES) OMITTED AND INSERT "?" FOR UNKOWN CHARACTERS

H1

#_ _ _ _ _ _

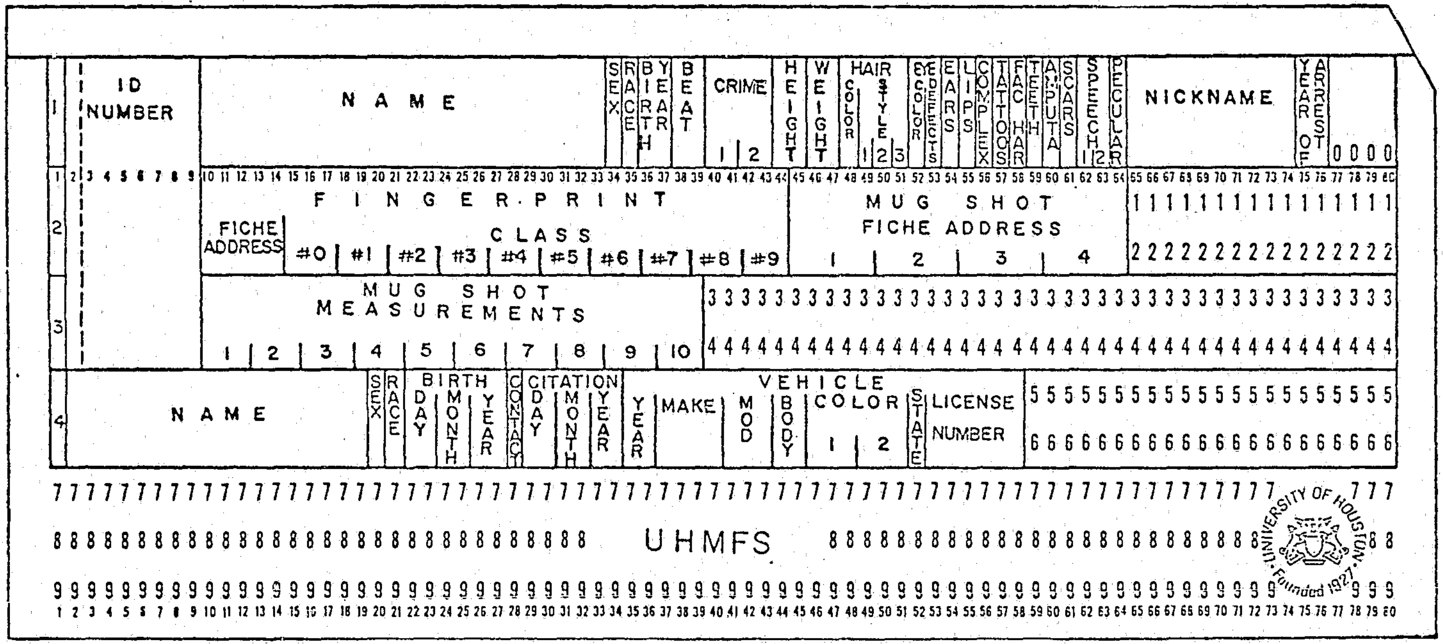


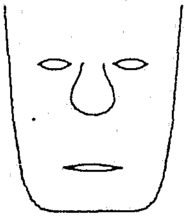
Figure A-4

Update Specification Record

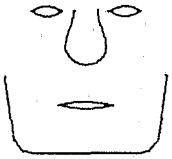
Card Format

73

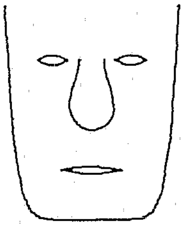
HPDMW 90



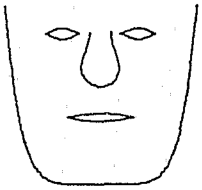
HPDMW 89



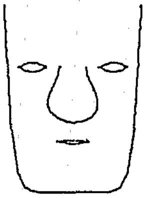
HPDMW 88



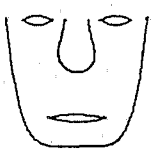
HPDMW 87



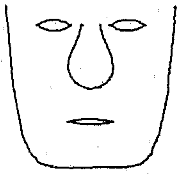
HPDMW 86



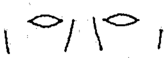
HPDMW 85



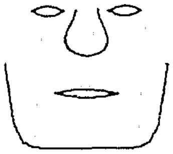
HPDMW 84



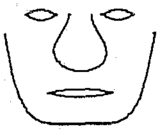
HPDMW 83



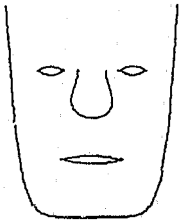
HPDMW 100



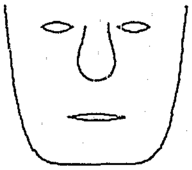
HPDMW 99



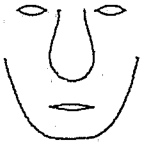
HPDMW 98



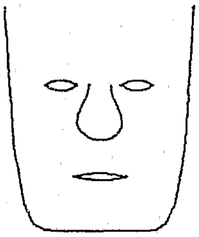
HPDMW 97



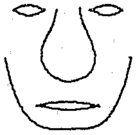
HPDMW 96



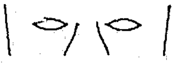
HPDMW 95



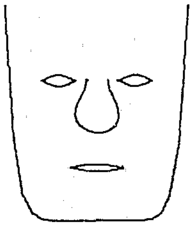
HPDMW 94



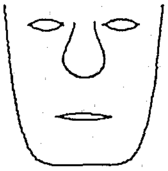
HPDMW 93



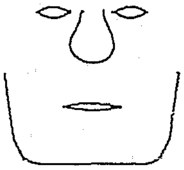
HPDMW 110



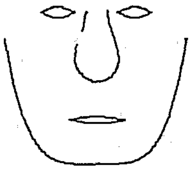
HPDMW 109



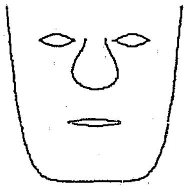
HPDMW 108



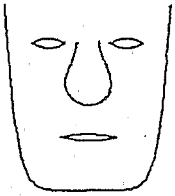
HPDMW 107



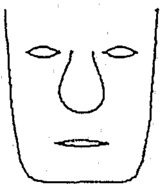
HPDMW 106



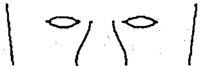
HPDMW 105



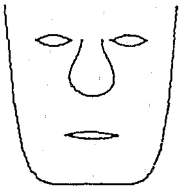
HPDMW 104



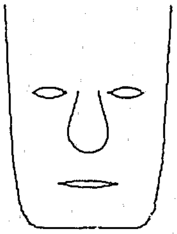
HPDMW 103



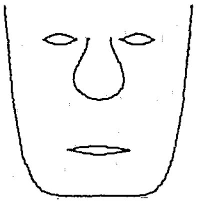
HPDMW 120



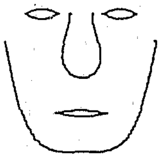
HPDMW 119



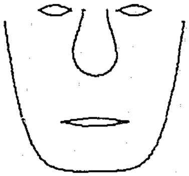
HPDMW 118



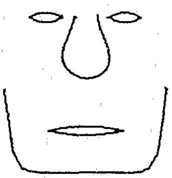
HPDMW 117



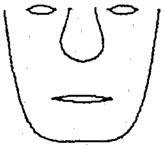
HPDMW 116



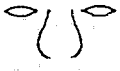
HPDMW 115



HPDMW 114



HPDMW 113



HP

HPD

HPD

HPD

HPD

HPD

HPD

HPDM

UNIVERSITY OF HOUSTON
CULLEN COLLEGE OF ENGINEERING
150 FORMAT(13X
6
12

END