NOAA Technical Memorandum NMFS

JANUARY 1994

REPORT ON CETACEAN AERIAL SURVEY DATA COLLECTED BETWEEN THE YEARS OF 1974 AND 1982

Timothy Lee

NOAA-TM-NMFS-SWFSC-195

U.S. DEPARTMENT OF COMMERCE
National Oceanic and Atmospheric Administration
National Marine Fisheries Service
Southwest Fisheries Science Center

NOAA Technical Memorandum NMFS

The National Oceanic and Atmospheric Administration (NOAA), organized in 1970, has evolved into an agency which establishes national policies and manages and conserves our oceanic, coastal, and atmospheric resources. An organizational element within NOAA, the Office of Fisheries is responsible for fisheries policy and the direction of the National Marine Fisheries Service (NMFS).

In addition to its formal publications, the NMFS uses the NOAA Technical Memorandum series to issue informal scientific and technical publications when complete formal review and editorial processing are not appropriate or feasible. Documents within this series, however, reflect sound professional work and may be referenced in the formal scientific and technical literature.

NOAA Technical Memorandum NMFS

This TM series is used for documentation and timely communication of preliminary results, interim reports, or special purpose information. The TMs have not received complete formal review, editorial control, or detailed editing.

JANUARY 1994

REPORT ON CETACEAN AERIAL SURVEY DATA COLLECTED BETWEEN THE YEARS OF 1974 AND 1982

Timothy Lee

La Jolla Laboratory, SWFSC
National Marine Fisheries Service, NOAA
P.O. Box 271
La Jolla, California 92038-0271

NOAA-TM-NMFS-SWFSC-195

U.S. DEPARTMENT OF COMMERCE
Ronald H. Brown, Secretary
National Oceanic and Atmospheric Administration
D. James Baker, Under Secretary for Oceans and Atmosphere
National Marine Fisheries Service
Rolland A. Schmitten, Assistant Administrator for Fisheries

REPORT ON CETACEAN AERIAL SURVEY DATA COLLECTED BETWEEN THE YEARS OF 1974 AND 1982.

Timothy Lee

National Marine Fisheries Service, NOAA
Southwest Fisheries Science Center
P.O. Box 271
La Jolla, California 92038

INTRODUCTION

In this paper I summarize the aerial survey data collected by the Southwest Fisheries Science Center (SWFSC) between the years of 1974 and 1982. This project was undertaken to allow comparisons to be made between cetacean distribution and abundance determined from older aerial surveys and the ones conducted more recently. To facilitate this comparison, I converted the data from the previous aerial surveys to the format that is being used currently for offshore surveys conducted by the Southwest Fisheries Science Center. The FORTRAN programs that accomplish this conversion, AIR74 and AIRYR, are listed in Appendices 1 and 2. I also rewrote the FORTRAN program ABUND, which was originally written for ship survey data, to produce summary statistics for aerial surveys (Program AIRBUND). In addition I changed the CRUZPLT program into AIRPLOT, a program that plots the data from the aerial surveys.

METHODS

The data collected before 1982 are stored in two different formats. The 1974 data are stored in one format (Appendix 3), and the data collected after 1974 are stored in a separate format (Appendix 4). Both formats have the sighting and transect record data separated into two files. The transect record contains information about position and speed of the aircraft, Beaufort sea state, and altitude of the plane, etc. The sighting record contains information about the position of marine mammals, the distance from the track line to the animals, species identification, "best" "high" and "low" school size estimates, etc. The new format has the sighting and transect information integrated into one file (Appendix 5). Each data line has a letter code in the fourth column that describes the type of event. For example, a resume effort event has an "R" code followed by the time and position effort was resumed.

Each of the species or stocks has a two-digit number assigned to it by the SWFSC (Appendix 6). This number is used on the observer data forms as a species identification code. Once a school of marine mammals has been spotted, the observers identify the species as best they can and estimate school size. It is not

always possible to identify the animals to the species or subspecies level, so there are also codes for "unidentified" species sightings. For example, code "05" is used for unidentified Delphinus, meaning the observers were unable to discern whether this was a school of long-beaked or short-beaked Delphinus.

Not all data were recorded for all years. For example, percent glare and declination angle were not recorded from 1974 to 1982. In addition, there were no independent observers prior to 1991.

Because the 1974 survey was the first SWFSC attempt at surveying for marine mammals from the air, it was largely a prototype experiment. The effort files are missing crucial data and it is not possible to determine whether the sightings were made on or off-effort. Because of this, the effort data were not converted and no plots of effort were made. Instead only the sightings were plotted.

There was no accurate key available for the 1974 data format. This made the process of converting the sighting data slow and laborious because the data had to be deciphered before it could be processed. In addition the sighting codes were much different from those used presently. First, the new data format uses letter codes for species rather than number codes. Secondly, certain codes were used that have no equivalent in the present format. For instance, the species code 4 was a code that represented "ETP Spotters and Spinners"; however, in the new format there are no mixed species codes. To compensate for this, mixed species codes were split into two separate codes during the conversion.

SUMMARY OF AVAILABLE DATA

In all there were five different aerial surveys conducted between 1974 and 1982.

1974 (Jan 26 - Feb. 14): This survey was conducted from a Gruman Goose aircraft that had been modified for use in aerial surveys. The plane flew at altitudes ranging from 1,000 to 1,500 feet and surveyed from Mazatlan, Mexico to Balboa, Panama searching for schools of dolphins. The survey was coordinated with a ship survey being conducted concurrently from the DAVID STARR JORDAN.

1977 (Jan 22 - June 21): This survey was conducted from two different aircraft. Phases one and two were conducted from a P2V-7 (SP-2H) Neptune anti-submarine patrol bomber, and phases three and four were conducted from a PBY-6A Catalina amphibious Navy patrol bomber. Phases one and two were conducted off the Pacific coast from Mazatlan, Mexico to Lima, Peru. Phase three surveyed

from San Diego, CA to San Jose, Costa Rica. Phase four surveyed around the Hawaiian Islands. The purpose of this survey was to search for dolphin schools with methods that were compatible with line-transect analysis. Transect lines are illustrated in Figure 1.

1979 (Jan 22 - Apr. 25): This survey was conducted from a PBY-5A Catalina, a modified amphibious patrol aircraft. The aircraft surveyed along the Pacific between Puerto Vallarta, Mexico and Lima, Peru searching for schools of dolphins. The survey was coordinated with two ship surveys being conducted during the same time interval by the DAVID STARR JORDAN and the TOWNSEND CROMWELL. The purpose of coordinating the three surveys was to quantify the bias from ship surveys caused by dolphins avoiding the ship. Transect lines are illustrated in Figure 2.

1981 (March 7 - April 5): This survey was conducted from a Beech AT11 aircraft that surveyed near Liberia, Costa Rica. The purpose was to investigate how estimates of marine mammal density and detectability were affected by observer experience, sea state, glare, and cloud cover. Transect lines are illustrated in Figure 3.

1982 (April 13 - April 15): This flight was conducted from a Beechcraft E18S aircraft in the Southern California Bight. There were two purposes for this survey. The first was to duplicate the flight transects that were used by Dohl et al (1978). The second was to gather data on the relative abundance of the short-finned pilot whale. Transect lines are illustrated in Figure 4.

RESULTS

Aerial survey track lines and distributions of sightings of the common cetacean species are illustrated in Figures 1 to 14. Encounter rates of the number of groups seen and the estimated number of individuals seen of each species is given for each cruise in Tables 1 to 4.

ACKNOWLEDGEMENTS

I would like to give special thanks to Jim Carretta and Chuck Oliver for their help with understanding the data formats. Thanks also to Jay Barlow and Tim Gerrodette for their assistance with programming.

Table 1. Number of sightings and number of individuals seen while on effort during the 1977 aerial survey. Sighting rates per nautical mile are based on these numbers divided by the number of on-effort survey miles. SWFSC codes are in parentheses.

1977 AERIAL SURVEY

	SIGHTING	ANIMALS	SIGHT/NM	ANMLS/NM
Stenella attenuata A (02)*	15	3278	0.00046	0.09969
Stenella longirostris (03)	4	434	0.00012	0.01320
Unidentified Delphinus (05)	17	3028	0.00052	0.09208
Stenella attenuata g. (06)*	2	214	0.00006	0.00651
S. longirostris o. (10)*	5	760	0.00015	0.02311
S. coeruleoalba (13)	5	544	0.00015	0.01654
Tursiops truncatus (18)	16	302	0.00049	0.00918
Grampus griseus (21)	83	2229	0.00252	0.06779
Pseudorca crassidens (33)	2	4	0.00006	0.00012
Unident. pilot whale (34)	5	13	0.00015	0.00040
Physeter macrocephalus (46)	19	76	0.00058	0.00231
Unidentified ziphiid (49)	3	5	0.00009	0.00015
Unidentified rorqual (70)	3	3	0.00009	0.00009
B. acutorostrata (71)	1	1	0.00003	0.00003
B. edeni (72)	1	1	0.00003	0.00003
B. musculus (75)	1	1	0.00003	0.00003
M. novaeangliae (76)	1	1	0.00003	0.00003
Unidentified delfinid (77)	76	2983	0.00231	0.09072
Unidentified sm. whale (78)	36	923	0.00109	0.02807
Unidentified lg. whale (79)	17	23	0.00052	0.00070
S. longirostris c. (88)*	1	980	0.00003	0.02980

Total Distance On Effort: 32,883 NMI

* Stenella attenuata subsp. A (offshore spotted dolphin)
* Stenella attenuata graffmani
* S. longirostris orientalis
* S. longirostris centroamericana

Table 2. Number of sightings and number of individuals seen while on effort during the 1979 aerial survey. Sighting rates per nautical mile are based on these numbers divided by the number of on-effort survey miles. SWFSC codes are in parentheses.

	1979 AERIAL SURVEY			
	SIGHTING	ANIMALS	SIGHT/NM	ANMLS/NM
Stenella attenuata A (02)*	8	3470	0.00052	0.22585
Unidentified Delphinus (05)	31	6941	0.00202	0.45177
Stenella attenuata g. (06)*	5	837	0.00033	0.05448
S. longirostris o. (10)*	6	3584	0.00039	0.23327
S. coeruleoalba (13)	6	533	0.00039	0.03469
Steno bredanensis (15)	1	54	0.00007	0.00351
Tursiops truncatus (18)	10	121	0.00065	0.00788
Grampus griseus (21)	76	1162	0.00495	0.07563
Lagenodelphis hosei (26)	1	60	0.00007	0.00391
Pseudorca crassidens (33)	1	7	0.00007	0.00046
Unident. pilot whale (34)	6	68	0.00039	0.00443
Orcinus orca (37)	1	16	0.00007	0.00104
Physeter macrocephalus (46)	29	323	0.00189	0.02102
Unidentified ziphiid (49)	11	24	0.00072	0.00156
Mesoplodon sp. (51)	1	2	0.00007	0.00013
Ziphius cavirostris (61)	1	3	0.00007	0.00020
Unidentified rorqual (70)	8	11	0.00052	0.00072
B. edeni (72)	1	1	0.00007	0.00007
Unidentified delfinid (77)	142	5187	0.00924	0.33761
Unidentified sm. whale (78)	26	438	0.00169	0.02851
Unidentified lg. whale (79)	15	19	0.00098	0.00124

Total Distance On Effort: 15,364 NMI

* Stenella attenuata subsp. A (offshore spotted dolphin)
* Stenella attenuata graffmani
* S. longirostris orientalis

Table 3. Number of sightings and number of individuals seen while on effort during the 1981 aerial survey. Sighting rates per nautical mile are based on these numbers divided by the number of on-effort survey miles. SWFSC codes are in parentheses.

	1981 AERIAL SURVEY			
	SIGHTING	ANIMALS	SIGHT/NM	ANMLS/NM
Stenella longirostris (03)	10	4583	0.00145	0.66267
Stenella attenuata g. (06)*	6	1669	0.00087	0.24132
S. coeruleoalba (13)	2	322	0.00029	0.04656
Grampus griseus (21)	29	271	0.00419	0.03918
Pseudorca crassidens (33)	3	96	0.00043	0.01388
Unidentified ziphiid (49)	3	7	0.00043	0.00101
Unidentified delfinid (77)	277	34367	0.04005	4.96920
Unidentified sm. whale (78)	27	556	0.00390	0.08039
Unidentified lg. whale (79)	1	1	0.00014	0.00014
Unid S. attenuata (90)	16	4938	0.00231	0.71400
Unidentified cetacean (96)	3	33	0.00043	0.00477

Total Distance On Effort: 6,916 NMI

* Stenella attenuata graffmani

Table 4. Number of sightings and number of individuals seen while on effort during the 1982 aerial survey. Sighting rates per nautical mile are based on these numbers divided by the number of on-effort survey miles. SWFSC codes are in parentheses.

	1982 AERIAL SURVEY			
	SIGHTING	ANIMALS	SIGHT/NM	ANMLS/NM
Tursiops truncatus (18)	6	19	0.00466	0.01475
Grampus griseus (21)	10	65	0.00776	0.05047
Phocoenoides dalli (44)	8	51	0.00621	0.03960
Unidentified delfinid (77)	4	20	0.00311	0.01553
Unidentified sm. whale (78)	3	10	0.00233	0.00776
Zalophus californianus (91)	5	20	0.00388	0.01553
Unidentified pinniped (97)	14	18	0.01087	0.01398
Unidentified whale (98)	2	2	0.00155	0.00155
Unidentified cetacean (99)	1	17	0.00078	0.01320

Total Distance On Effort: 1,288 NMI

*This flight used a modified version of the standard SWFSC code table.

Figure 1.

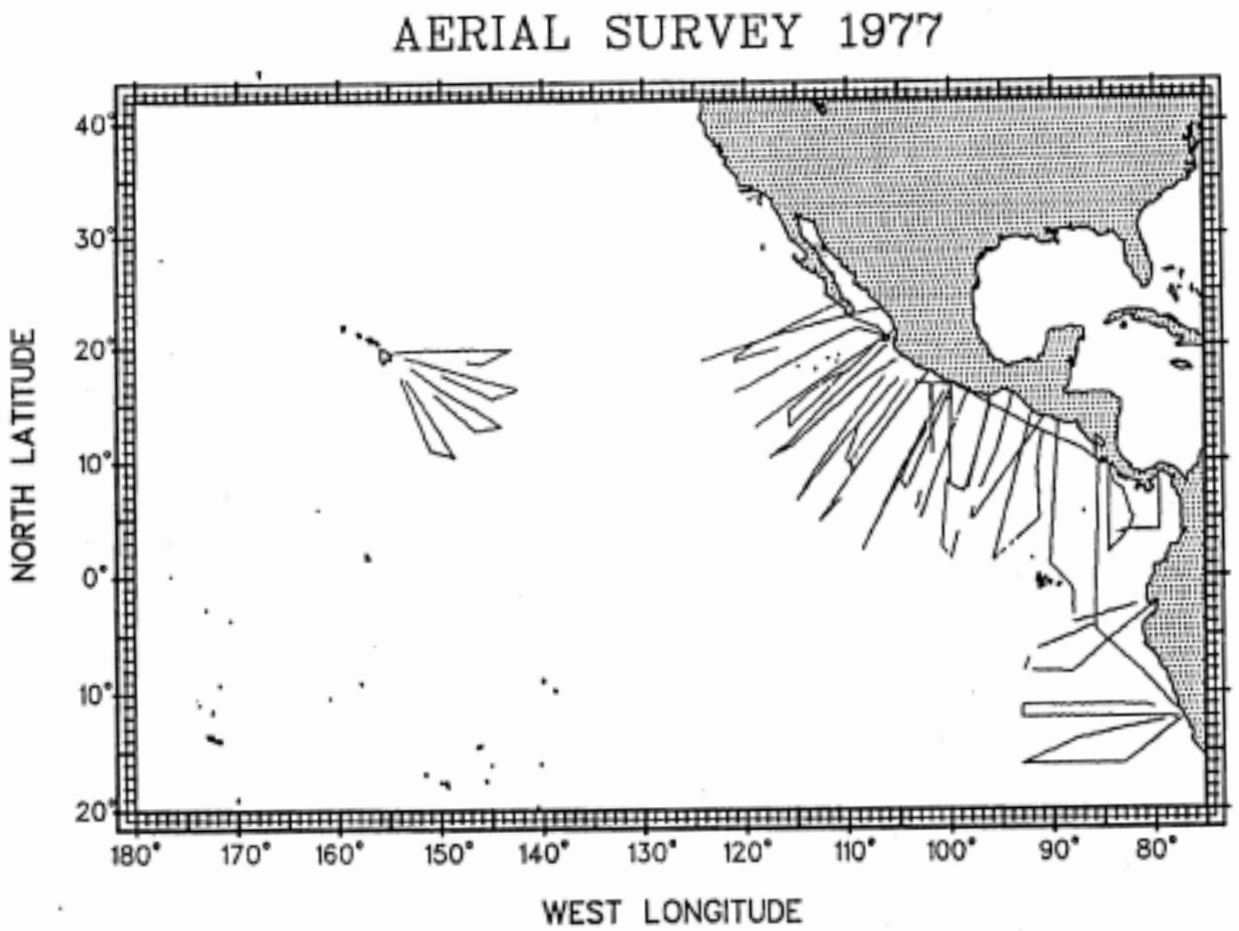

AERIAL SURVEY 1977
NORTH LATITUDE
40°
30°
20°
10°
0°
10°
20°
180°
170°
160°
150°
140°
130°
120°
110°
100°
90°
80°
WEST LONGITUDE

Figure 2.

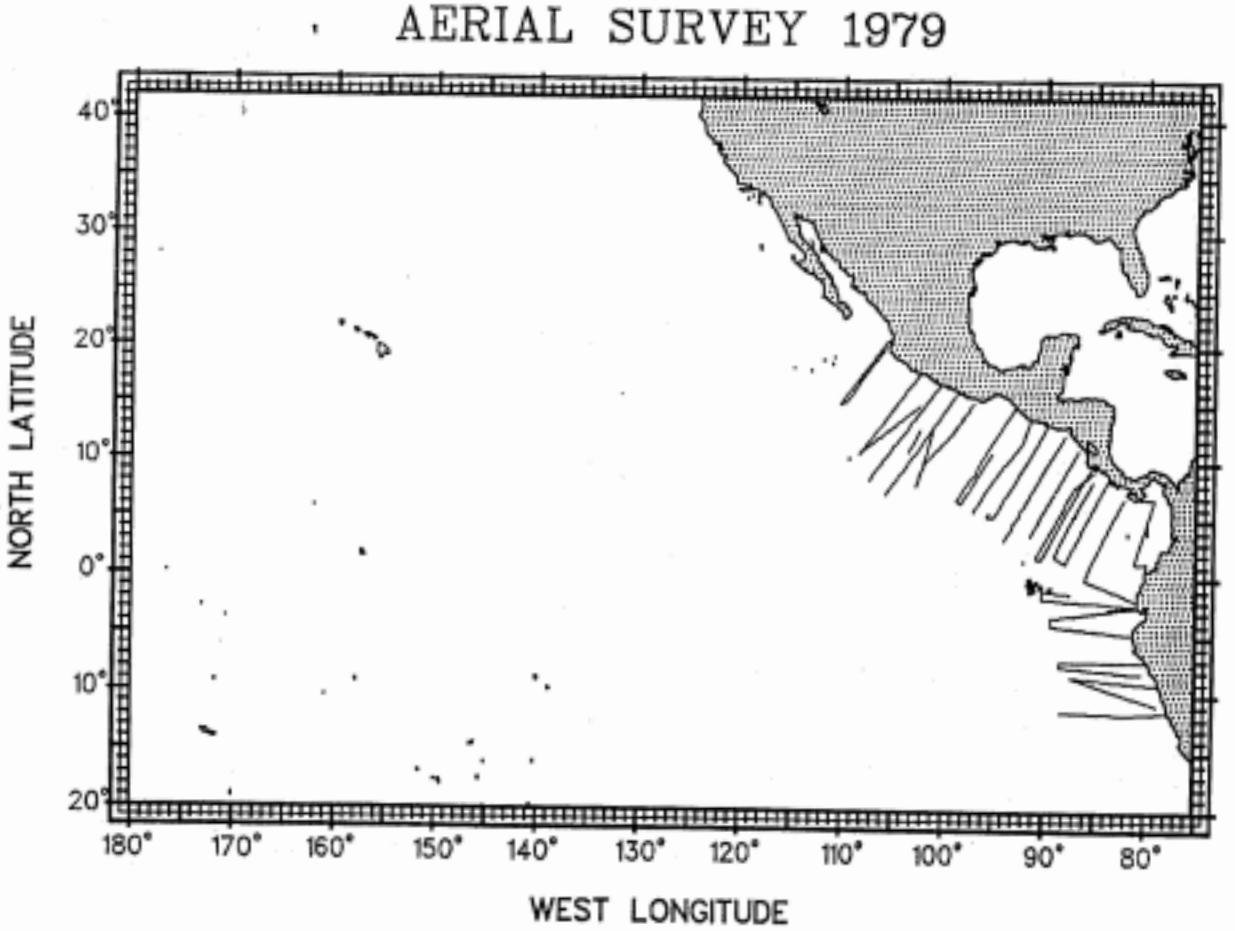

AERIAL SURVEY 1979
NORTH LATITUDE
40°
30°
20°
10°
0°
10°
20°
180°
170°
160°
150°
140°
130°
120°
110°
100°
90°
80°
WEST LONGITUDE

Figure 3.

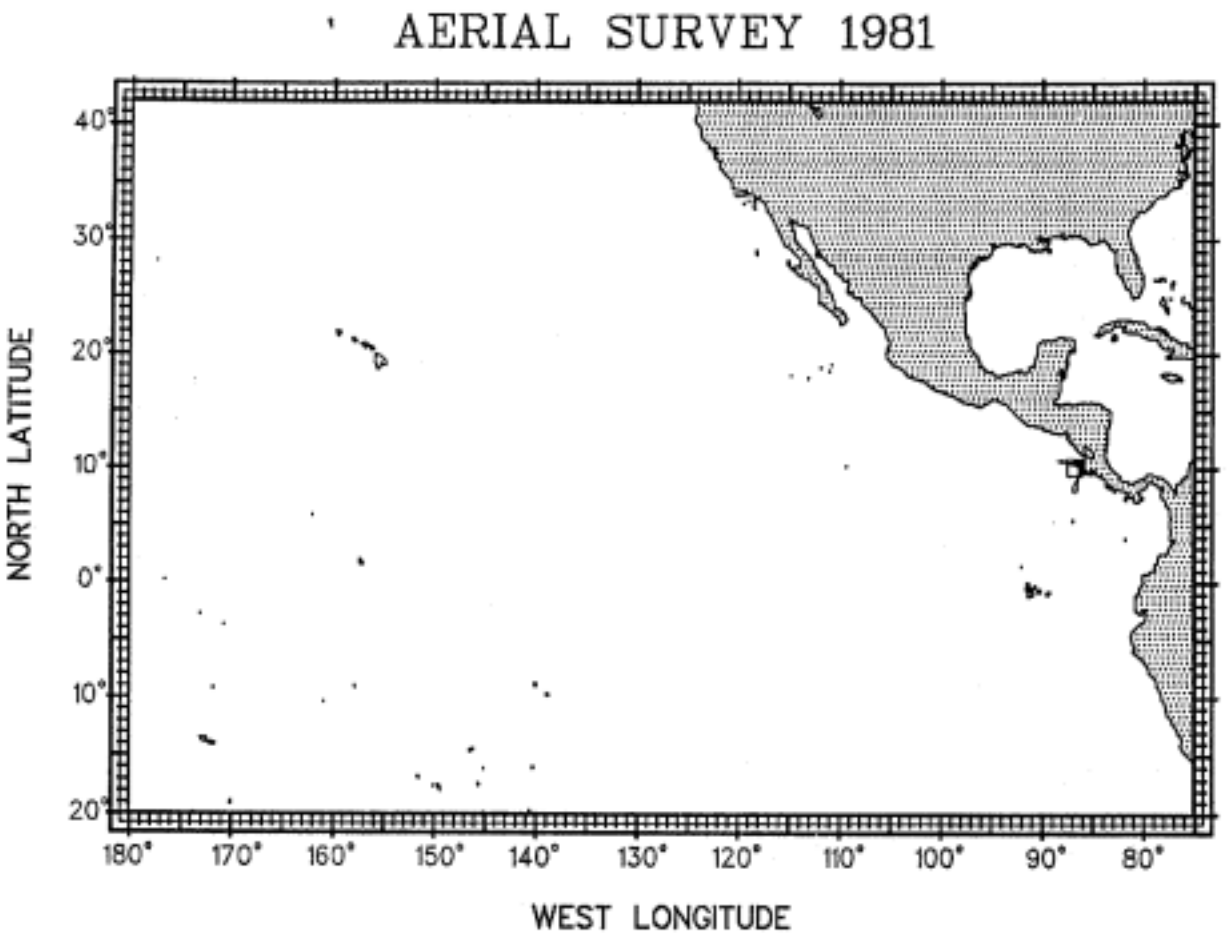

AERIAL SURVEY 1981
NORTH LATITUDE
40°
30°
20°
10°
0°
10°
20°
180°
170°
160°
150°
140°
130°
120°
110°
100°
90°
80°
WEST LONGITUDE

Figure 4.

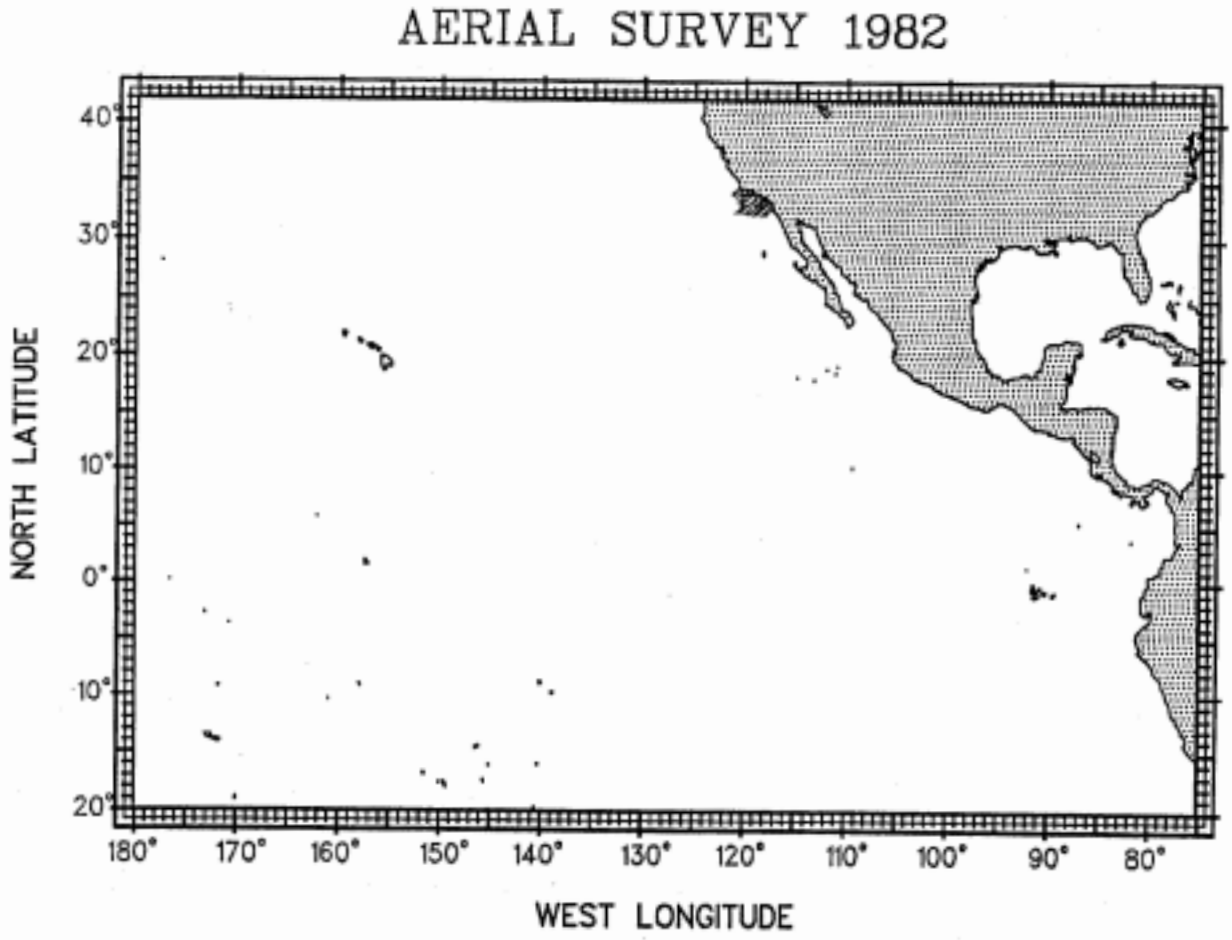

AERIAL SURVEY 1982
NORTH LATITUDE
40°
30°
20°
10°
0°
10°
20°
180°
170°
160°
150°
140°
130°
120°
110°
100°
90°
80°
WEST LONGITUDE

Figure 5.

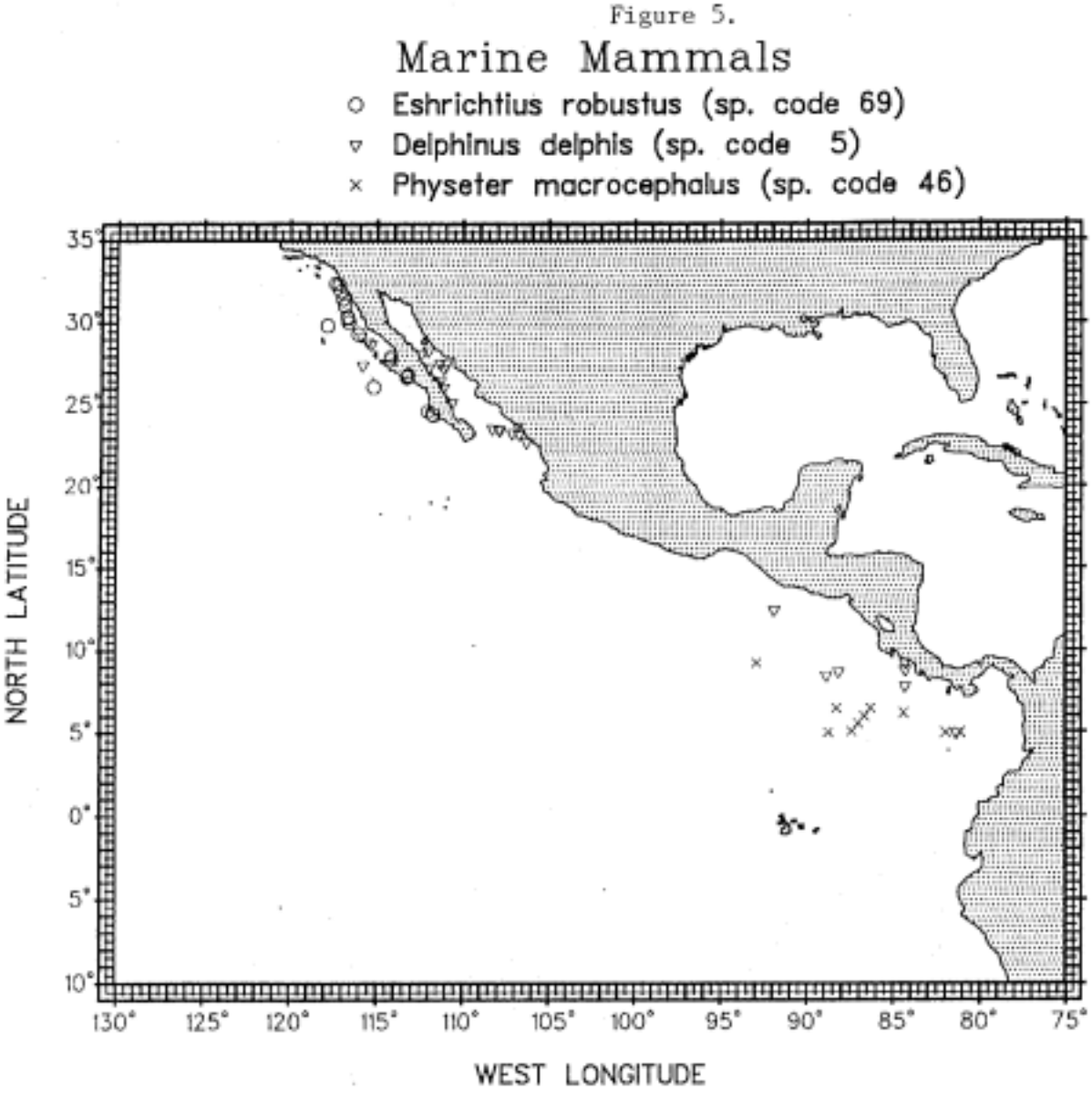

Marine Mammals
Eshrichtius robustus (sp. code 69)
Delphinus delphis (sp. code 5)
Physeter macrocephalus (sp. code 46)
NORTH LATITUDE
35°
30°
25°
20°
15°
10°
5°
0°
5°
10°
130°
125°
120°
115°
110°
105°
100°
95°
90°
85°
80°
75°
WEST LONGITUDE

Figure 6.

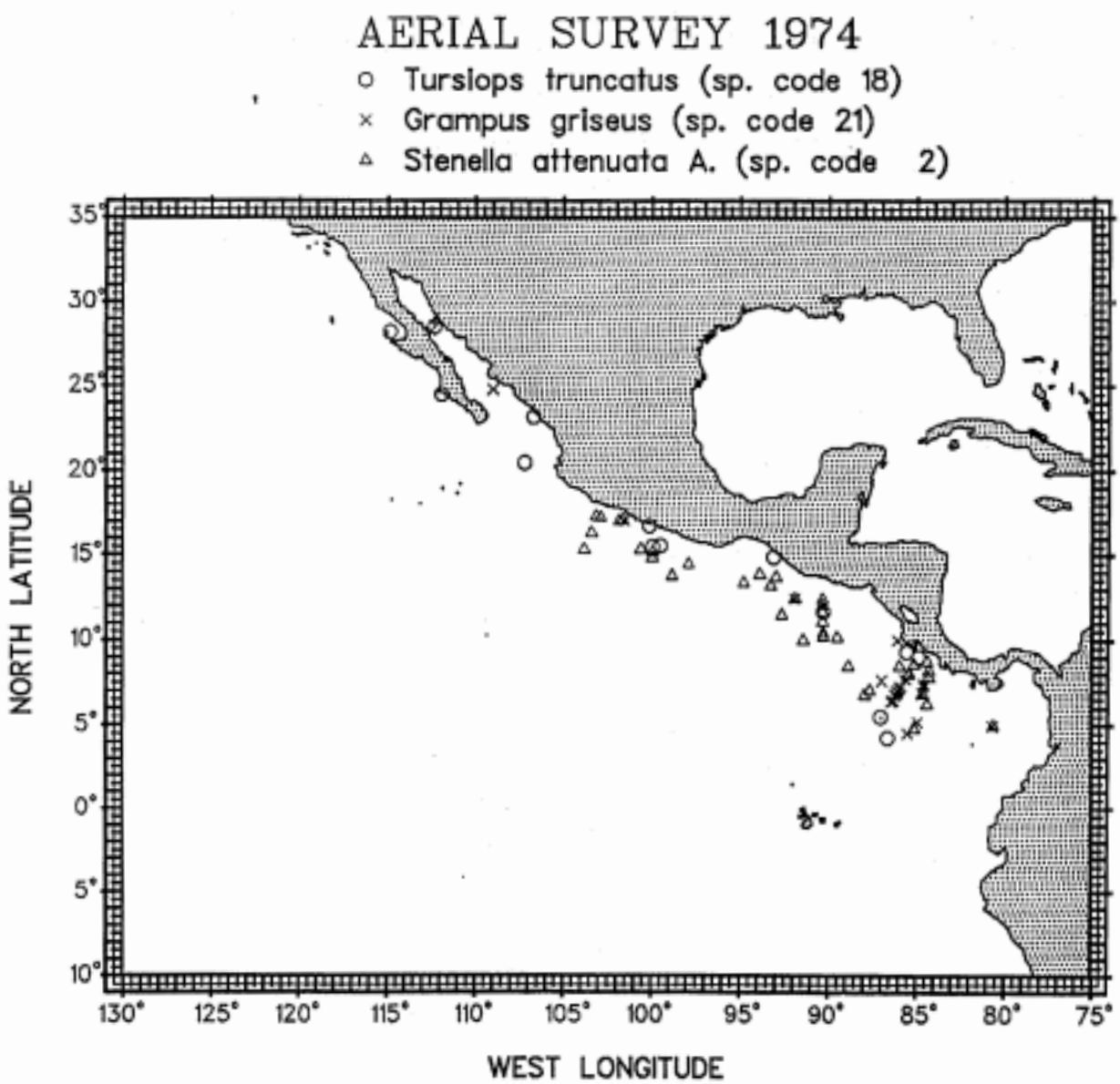

AERIAL SURVEY 1974
Tursiops truncatus (sp. code 18)
Grampus griseus (sp. code 21)
Stenella attenuata A. (sp. code 2)
NORTH LATITUDE
35°
30°
25°
20°
15°
10°
5°
0°
5°
10°
130°
125°
120°
115°
110°
105°
100°
95°
90°
85°
80°
75°
WEST LONGITUDE

Figure 7.

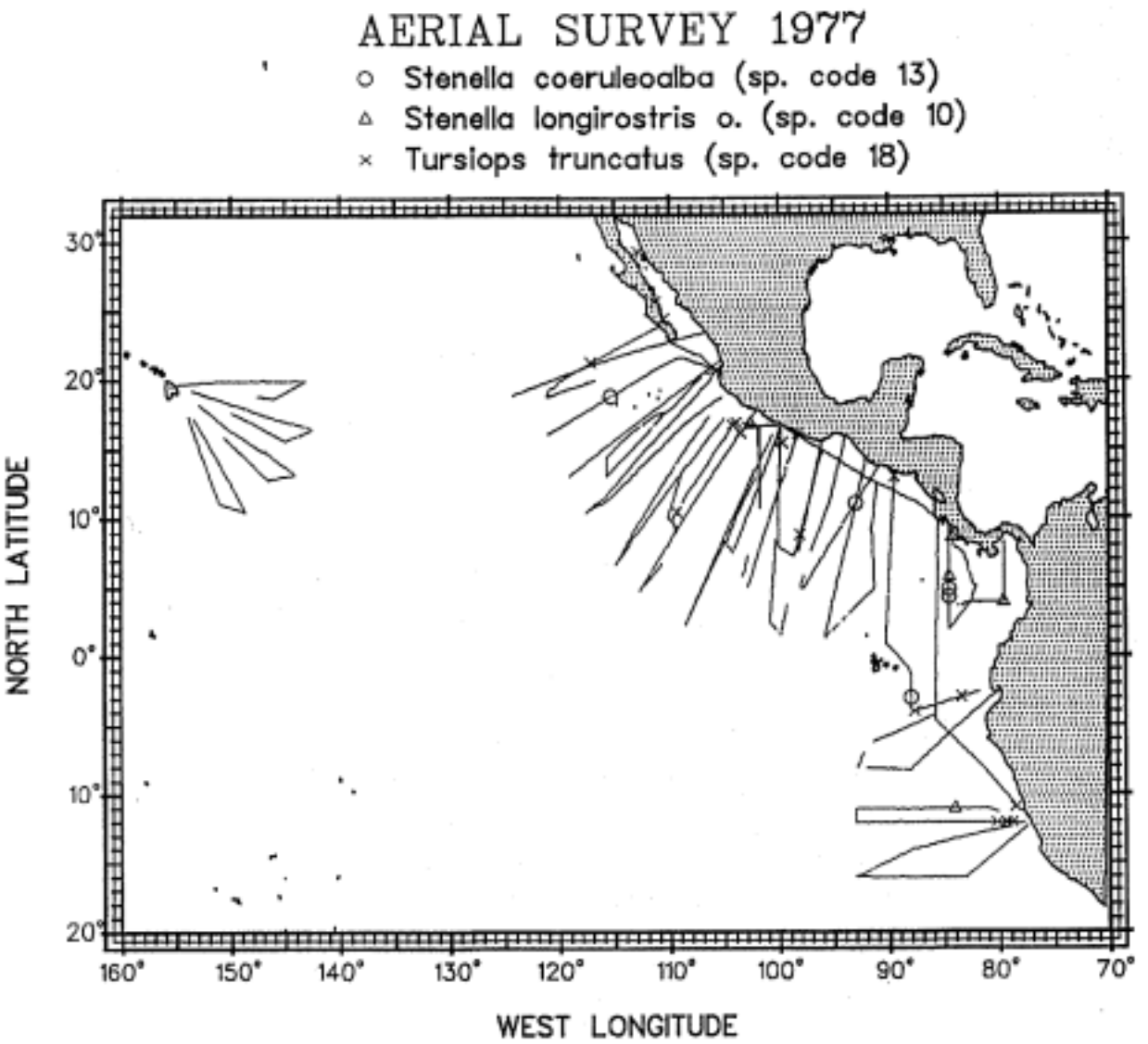

AERIAL SURVEY 1977
Stenella coeruleoalba (sp. code 13)
Stenella longirostris o. (sp. code 10)
Tursiops truncatus (sp. code 18)
NORTH LATITUDE
30°
20°
10°
0°
10°
20°
160°
150°
140°
130°
120°
110°
100°
90°
80°
70°
WEST LONGITUDE

Figure 8.

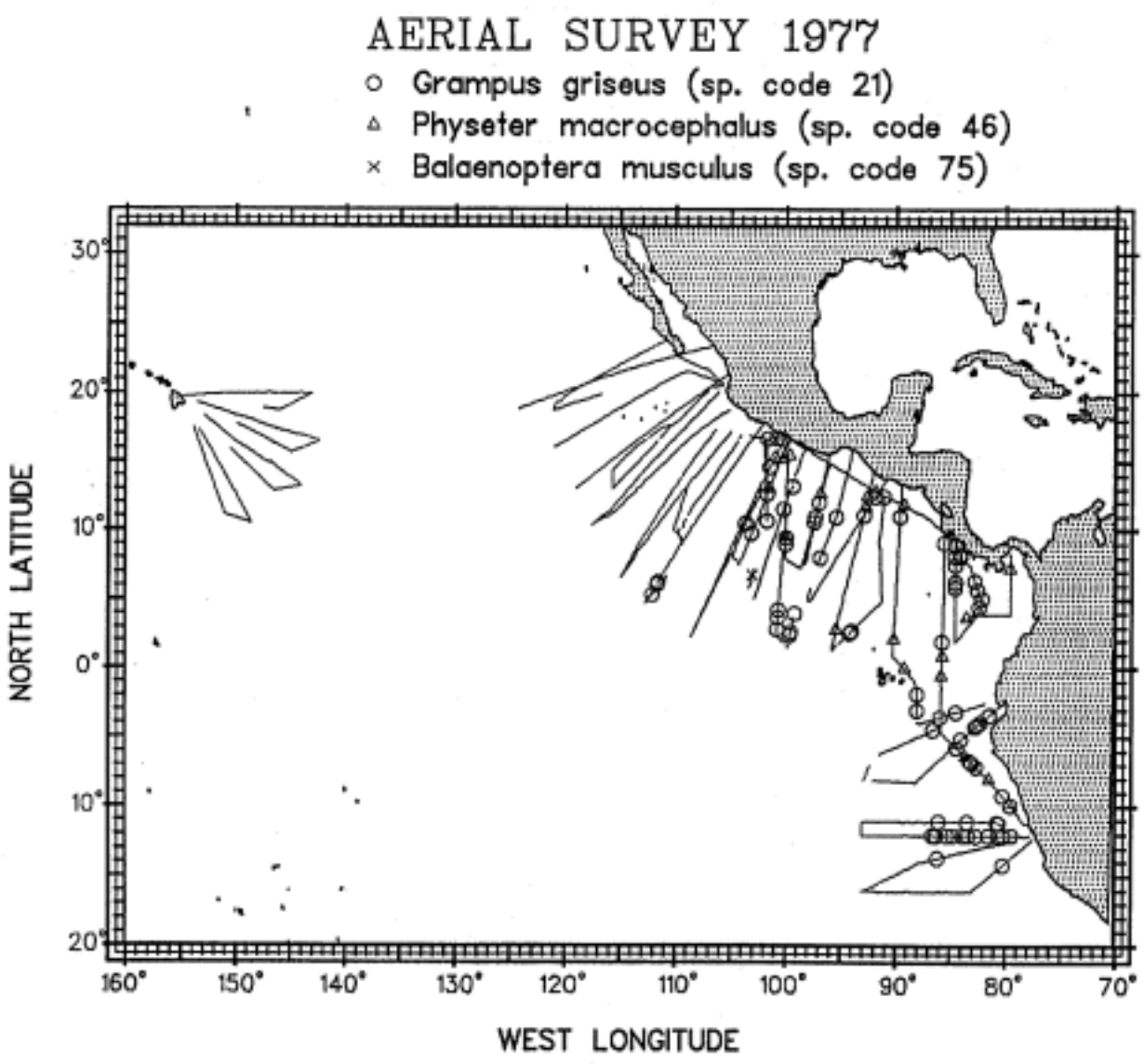

AERIAL SURVEY 1977
Grampus griseus (sp. code 21)
Physeter macrocephalus (sp. code 46)
Balaenoptera musculus (sp. code 75)
NORTH LATITUDE
30°
20°
10°
0°
10°
20°
160°
150°
140°
130°
120°
110°
100°
90°
80°
70°
WEST LONGITUDE

Figure 9.

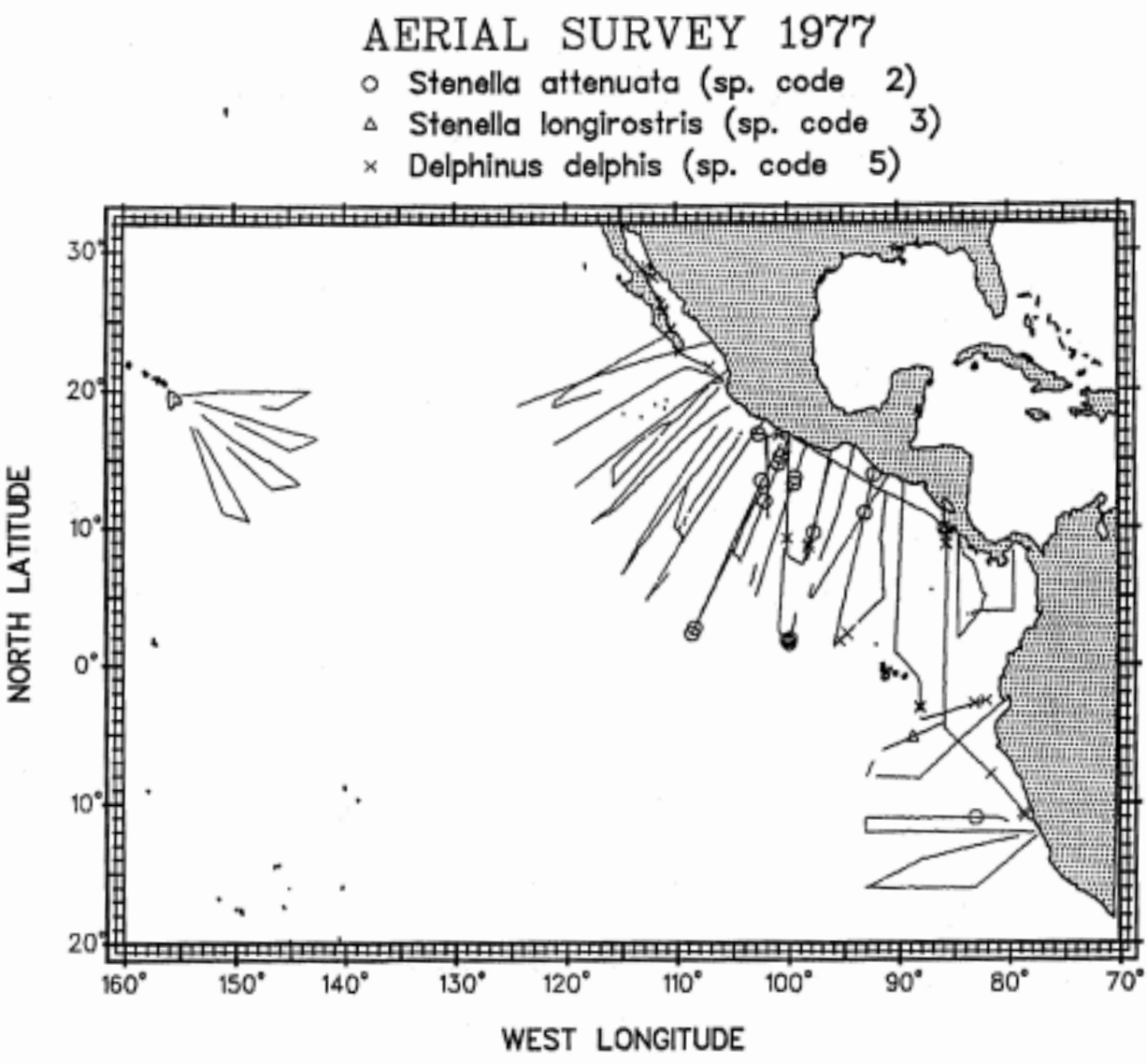

AERIAL SURVEY 1977
○ Stenella attenuata (sp. code 2)
△ Stenella longirostris (sp. code 3)
× Delphinus delphis (sp. code 5)
30°
20°
10°
0°
10°
20°
NORTH LATITUDE
160°
150°
140°
130°
120°
110°
100°
90°
80°
70°
WEST LONGITUDE

Figure 10.

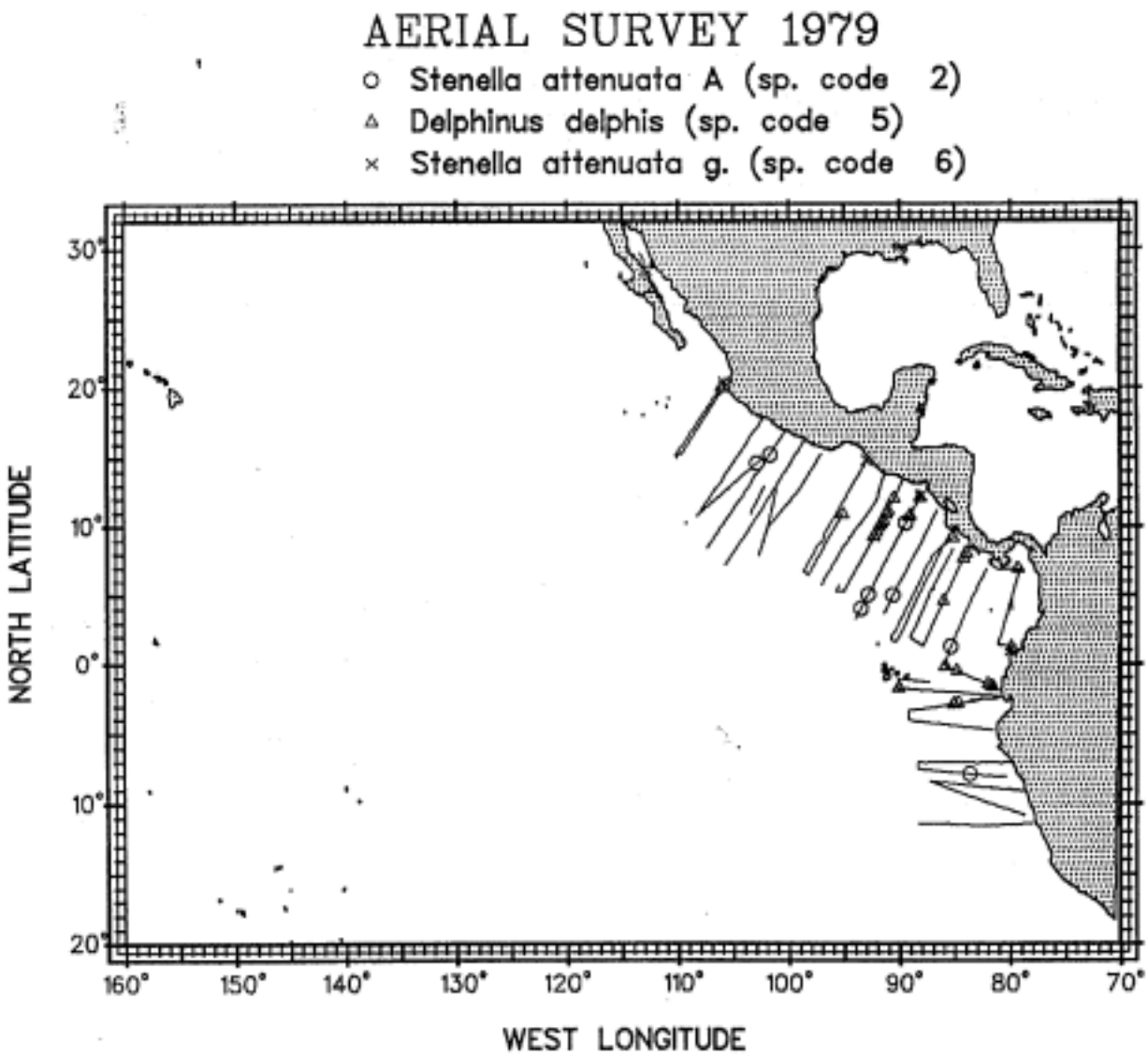

AERIAL SURVEY 1979
Stenella attenuata A (sp. code 2)
Delphinus delphis (sp. code 5)
Stenella attenuata g. (sp. code 6)
NORTH LATITUDE
30°
20°
10°
0°
10°
20°
160°
150°
140°
130°
120°
110°
100°
90°
80°
70°
WEST LONGITUDE

Figure 11.

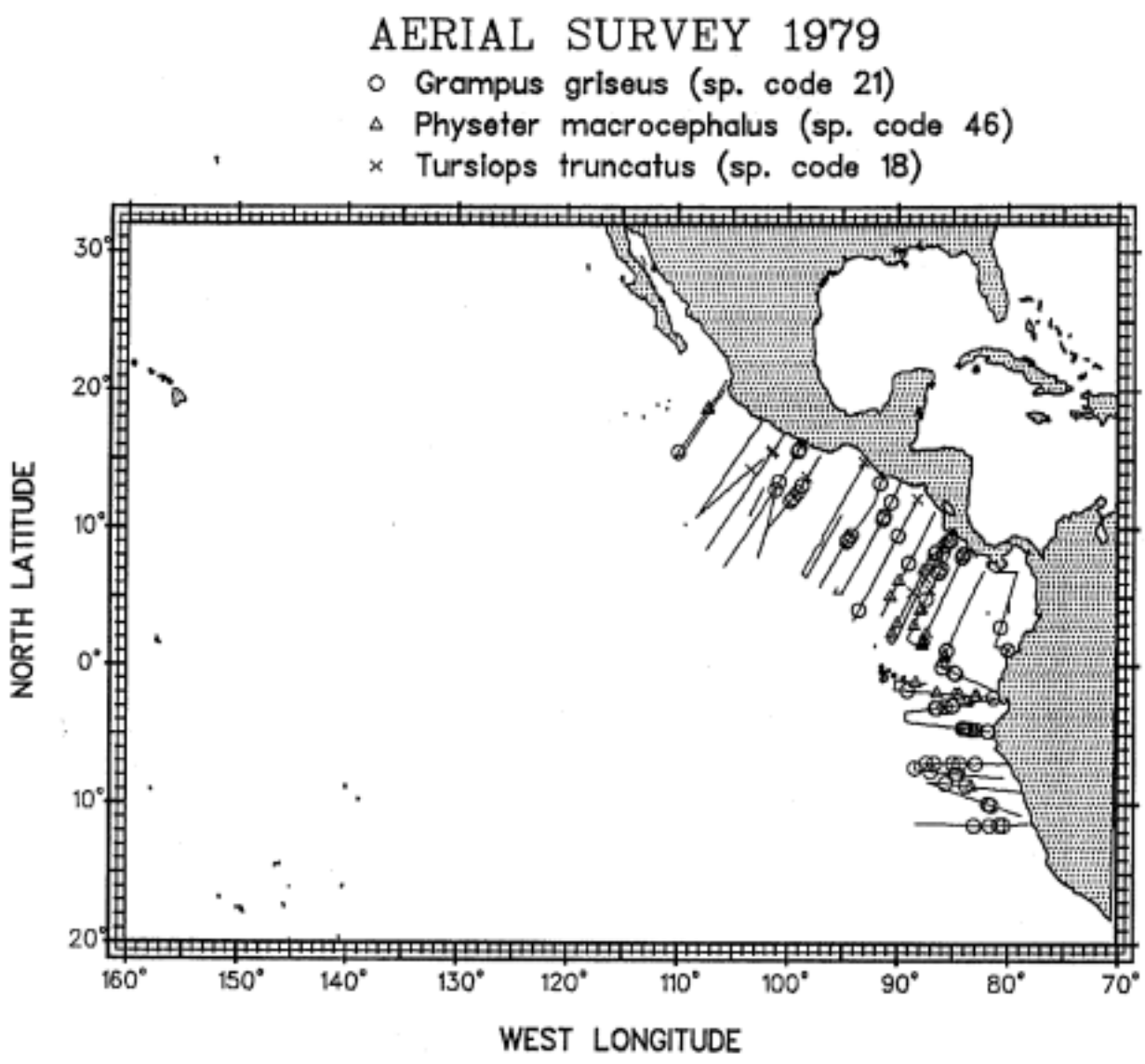

AERIAL SURVEY 1979
○ Grampus griseus (sp. code 21)
△ Physeter macrocephalus (sp. code 46)
× Tursiops truncatus (sp. code 18)
30°
20°
10°
0°
10°
20°
NORTH LATITUDE
160°
150°
140°
130°
120°
110°
100°
90°
80°
70°
WEST LONGITUDE

Figure 12.

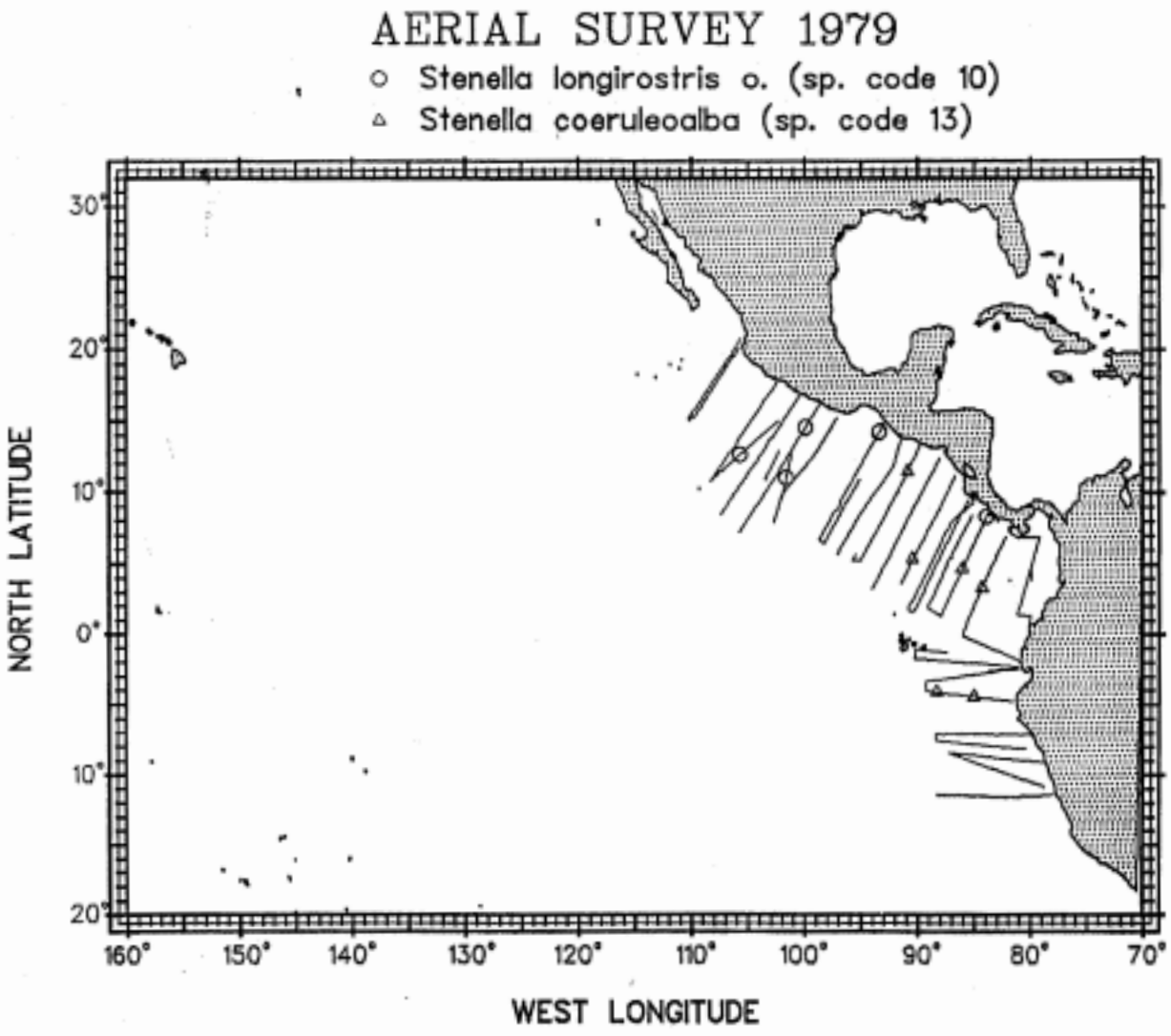

AERIAL SURVEY 1979
Stenella longirostris o. (sp. code 10)
Stenella coeruleoalba (sp. code 13)
NORTH LATITUDE
30°
20°
10°
0°
10°
20°
160°
150°
140°
130°
120°
110°
100°
90°
80°
70°
WEST LONGITUDE

Figure 13.

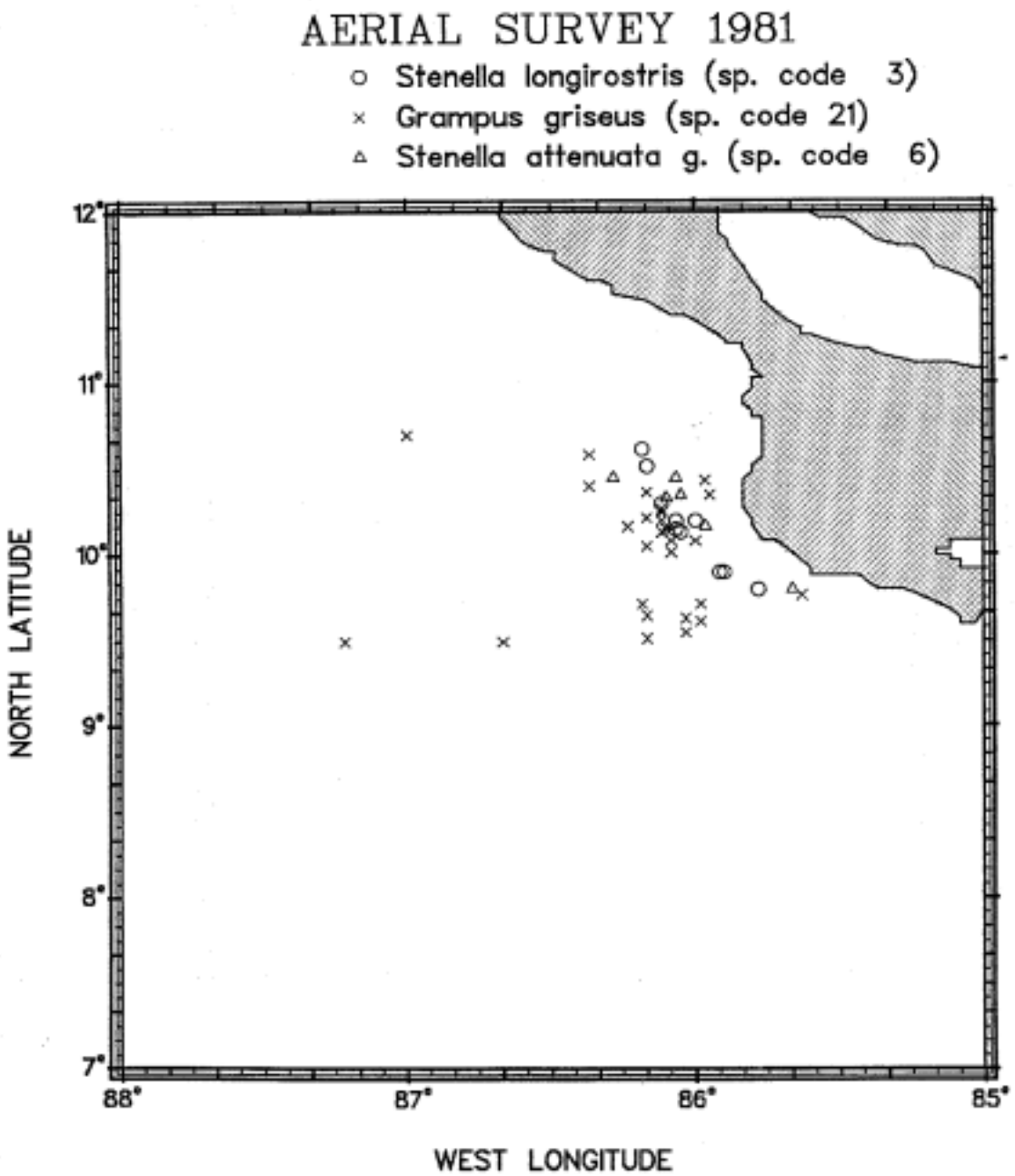
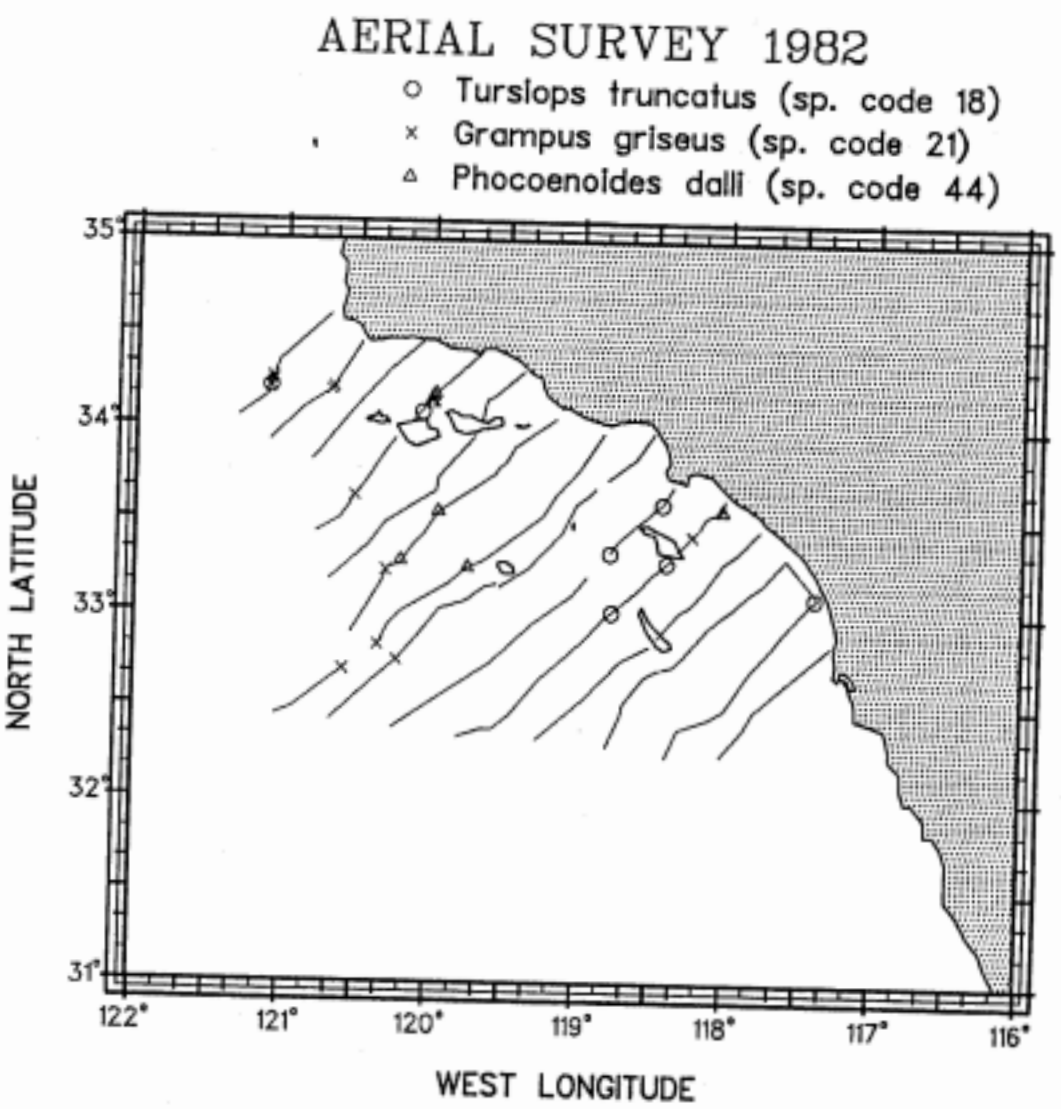

AERIAL SURVEY 1981
Stenella longirostris (sp. code 3)
Grampus griseus (sp. code 21)
Stenella attenuata g. (sp. code 6)
12°
11°
10°
9°
8°
7°
NORTH LATITUDE
88°
87°
86°
85°
WEST LONGITUDE

Figure 14.


AERIAL SURVEY 1982
Tursiops truncatus (sp. code 18)
Grampus griseus (sp. code 21)
Phocoenoides dalli (sp. code 44)
35°
34°
33°
32°
31°
NORTH LATITUDE
122°
121°
120°
119°
118°
117°
116°
WEST LONGITUDE

APPENDIX 1

```
************************************************************************
*
* This program takes the 1974 data and reformats into a data format    *
* similar to that used in the offshore surveys conducted currently (1993)*
*                                                                      *
* The effort data is unusable (I have been told) for several reasons   *
* One is the lack of accurate positions. The second is the fact that they*
* did not record when the plane was on or off effort. Instead they only *
* recorded the total number of minutes spent circling during a leg.    *
* Because of this, it is not possible to determine which sightings were *
* in fact on effort and which were off effort.                         *
*                                                                      *
*    Date June 10, 1993                                                *
*    Programmer: Timothy Lee                                           *
*                                                                      *
************************************************************************
      Program Air74_for

**DECLARATIONS**
      character pss*130,new*85,blankstring*85
      integer line
      logical MixSpp
      common new,pss,blankstring,line,MixSpp

**INITIALIZING EVERYTHING
      write(new,10)
      write(blankstring,10)
      line=0

**FORMAT STATEMENTS LINE #'S 10-30
10     format (100(' '))
11     format (a15,i3)
15     format (a)

**OPENING FILES FOR PROCESS AND STORAGE

      OPEN(UNIT=1,FILE='as74sq01.dat',STATUS='OLD')
      OPEN(UNIT=10,FILE='74.OUT', FORM='FORMATTED', STATUS='UNKNOWN')
      open(unit=11,file='74.err', form='formatted', status='unknown')

**READING IN DATA AND STORING IT IN VARIABLES

50     new(1:85)=blankstring(1:85)
      mixspp=.false.
      read(1,15, end=150) pss
      line=line+1


**PUTTING THE DATE AND TIME INTO THE HEAD OF STRING
      new(6:9)=pss(13:16)
      new(13:18)=pss(7:12)
****Position
      if (pss(22:22).eq.'1') new(20:20)='N'
```

```
      if (pss(22:22).eq.'2') new(20:20)='S'
      if (pss(29:29).eq.'1') new(30:30)='E'
      if (pss(29:29).eq.'2') new(30:30)='W'

      new(21:22)=pss(17:18)
      new(23:23)=':'
      new(24:25)=pss(19:20)
      new(26:26)='.'
      new(27:27)=pss(21:21)

      new(31:33)=pss(23:25)
      new(34:34)=':'
      new(35:36)=pss(26:27)
      new(37:37)='.'
      new(38:38)=pss(28:28)

*CHECKING TO SEE IF ANY OF THE CODES ARE MIXED SPECIES CODES
      if(pss(55:56).eq.' 4' .or. pss(55:56).eq.' 7' .or.
     * pss(55:56).eq.' 8'.or. pss(55:56).eq.' 9') MixSpp=.true.
      if(mixspp) write(11,11)'Mixspp line: ',line
*THIS IS A TEST TO SEE WHETHER ANY SPP CODE WAS RECORDED. IF NO SPP CODE
*WAS RECORDED -> FLAG AN ERROR AND SKIP THE LINE
      if(pss(55:56).eq.'  ') then
       write(11,11)'No Spp code line ',line
       goto 50
      endif

       call sight
        call length (new,lenth)
        write(10,15) new(1:lenth)

      call one
       call length (new,lenth)
       write (10,15) new(1:lenth)

      goto 50

150       end
*************************************************************************
      subroutine sight
**THIS SUBROUTINE CREATES AN S EVENT
      character pss*130,new*85,blankstring*85
      integer line
      logical MixSpp
      common new,pss,blankstring,line,MixSpp

*INITIALIZING THE END OF THE STRING
      new(41:85)=blankstring(41:85)

      new(4:4)='S'
*SIGHT #
      new(41:44)='  '//pss(5:6)
*OBSERVER CODE
      new(46:49)='  '//pss(30:31)
```

```
*DISTANCE
      new(51:54)=pss(34:35)//'.'//pss(36:36)
*PRIMARY OBSERVER Y/N
      new(56:59)='   Y'


       if (pss(65:66).eq.'  '.and. .not.MixSpp) then
*SPP CODE 1 = OBJECT SIGHTED
        new(61:64)='  '//pss(55:56)
       else if(MixSpp .and. pss(55:56).eq.' 4' .and. (pss(65:66)
    *.ne.' 2' .and. pss(65:66).ne.' 3') ) then
*SPP CODE 1 = 3 :UNID SPINNER
        new(61:64)='   3'
*SPP CODE 2 = 2 :OFFSHORE SPOTTER
        new(66:69)='   2'
*SPP CODE 3
        new(73:74)=pss(65:66)
       else if(MixSpp .and. pss(55:56).eq.' 7') then
*SPP CODE 1 = 3 :UNID SPINNER
        new(61:64)='   3'
*SPP CODE 2 = White Belly
        new(66:69)='   5'
*SPP CODE 3
        new(73:74)=pss(65:66)

       else if(MixSpp .and. pss(55:56).eq.' 8') then
*SPP CODE 1 = White Belly
        new(61:64)='   5'
*SPP CODE 2 = 2 :OFFSHORE SPOTTER
        new(66:69)='   2'
*SPP CODE 3
       if(pss(65:66).ne.' 2' .and. pss(65:66).ne.' 5')
     * new(73:74)=pss(65:66)

       else if(MixSpp .and. pss(55:56).eq.' 9') then
*SPP CODE 1 = 3 :UNID SPINNER
        new(61:64)='   3'
*SPP CODE 2 = White Belly Spinner
        new(66:69)='  11'
*SPP CODE 3
        new(73:74)=pss(65:66)

       else
*SPP CODE 1 = DOMINANT SPECIES
        new(61:64)='  '//pss(65:66)
*SPP CODE 2 = SECOND SPECIES
        new(66:69)='  '//pss(75:76)
*SPP CODE 3 = THIRD SPECIES
        new(71:74)='  '//pss(86:87)
       endif

      return
      end
```

```
**********************************************************************
      subroutine one
**THIS SUBROUTINE CREATES AN 'S' EVENT. THIS TYPE OF EVENT WAS CREATED
* TO FACILTATE INFORMATION THAT IS NOT FOUND IN THE NEW OFFSHORE AERIAL
* FORMAT
      character pss*130,new*85,blankstring*85
      real Percnt1,Percnt2,Percnt3
      logical MixSpp
      integer line
      common new,pss,blankstring,line,MixSpp

*INITIALIZING NEW PART OF STRING
      new(41:80)=blankstring(41:80)
      NumSpp=0

*CHECKING THE NUMBER OF SPECIES SIGHTED
      if(pss(55:56).ne.'  ') NumSpp=NumSpp+1
      if(pss(65:66).ne.'  ') NumSpp=NumSpp+1
      if(pss(75:76).ne.'  ') NumSpp=NumSpp+1
      if(pss(86:87).ne.'  ') NumSpp=NumSpp+1


      new(4:4)='1'
*OBS CODE
      new(41:44)='  '//pss(30:31)

       if (pss(65:66).eq.'  ' .and. .not.mixspp) then
        new(46:49)=pss(47:50)
        new(61:64)=' 100'
       else if ((pss(65:66).ne.'  '.and. .not.mixspp) .or. (pss(65:66)
    *.eq.' 2' .or. pss(65:66).eq.' 3' .or. pss(65:66).eq.' 5') ) then
        read(pss(47:50),'(f4.0)')total
        read(pss(57:60),'(f4.0)')SppCount1
        read(pss(67:70),'(f4.0)')SppCount2
        read(pss(77:80),'(f4.0)')SppCount3

        Percnt1=(SppCount1/total)*100
        Percnt2=(SppCount2/total)*100
        Percnt3=(SppCount3/total)*100
        tprcnt=Percnt1+Percnt2+Percnt3
         if(tprcnt.ge.99) then
          new(46:49)=pss(47:50)
          write(new(61:64),'(f4.1)')Percnt1
          write(new(66:69),'(f4.1)')Percnt2
          if (Percnt3.gt.0) write(new(71:74),'(f4.1)')Percnt3
         endif
       else if ((pss(65:66).ne.'  ') .and. (MixSpp)) then
          read(pss(47:50),'(f4.0)')total
          read(pss(57:60),'(f4.0)')SppCount1

          Percnt1=(SppCount1/total)*100
          new(46:49)=pss(47:50)
           if(Percnt1 .ge. .1) then
            write(new(71:74),'(f4.1)')Percnt1
```

```
          else
           write(new(71:74),'(f4.3)')Percnt1
          endif

        else
          new(46:49)=pss(47:50)
          new(63:64)='??'
          new(68:69)='??'

        endif


       return
       end
*************************************************************************
       SUBROUTINE  LENGTH(NEW,LENTH)
*
* This subroutine estimates the length of a character string excluding
*     the trailing blanks
       CHARACTER*85  NEW
        LENTH= 85
        DO 10 I=85,1,-1
         IF(NEW(I:I).EQ.'  ')THEN
          LENTH=I-1
         ELSE
          RETURN
         ENDIF
10       CONTINUE
       RETURN
       END
```

```
************************************************************************
* This program takes aerial survey data from the years of 1977,79,81, 82, *
* and converts them into the format that is being used for the offshore  *
* aerial surveys currently (1993).                                       *
*                                                                        *
* Written by Timothy Lee                                                 *
*          March 30, 1993                                                *
*                                                                        *
************************************************************************
      Program AIRYR_FOR

      character NEW*90,BLANKSTRING*90,  PREVC*1,NEW1*90,NEW2*90,CD2*1
      CHARACTER*30 OLD_SIGHT,OLD_EFF,NEW_COMB,  ERROR_OUT
      CHARACTER PES*500,PSS*250,  TIME_S*5,B$*2,DATE1*6,DATE2*6,CD1*1
      CHARACTER*90 PLINE,OLDP,OLDW,OLDA,OLDG
      CHARACTER PREVSYN*1,SYN*1
      LOGICAL NEWEFF,EFFORT
      INTEGER SNUM_S, SNUM_E, RO, COUNT
      COMMON NEW,B$,PSS,BLANKSTRING

      B$='  '
      WRITE(BLANKSTRING,17)
      COUNT=0
      DATE1='0'
      SYN='0'
      PREVSYN='0'
      EFFORT=.FALSE.

*MODULE FOR OPENING FILES*****************************************
      OPEN(UNIT=10,FILE='AIRYR.INP',FORM='FORMATTED',STATUS='OLD')
      PRINT*,'OPEN'

2     READ (10,'(A30)') OLD_SIGHT
      PRINT*,'OLDSIGHT  READ '//OLD_SIGHT
      READ (10,'(A30)') OLD_EFF
      PRINT*,'OLDEFF  READ '//OLD_EFF
      READ(10,'(A30)') ERROR_OUT
      PRINT*,'ERROR  FILE READ: '//ERROR_OUT
      READ (10,'(A30)') NEW_COMB
      PRINT*,'NEWCOMB  READ '//NEW_COMB

      OPEN(UNIT=1,FILE=OLD_SIGHT,STATUS='OLD')
      OPEN(UNIT=2,FILE=OLD_EFF,STATUS='OLD')
      OPEN(UNIT=11,FILE=ERROR_OUT,  FORM='FORMATTED',
     *     STATUS='UNKNOWN')
      OPEN(UNIT=12,FILE='NEW_EFF.OUT',  FORM='FORMATTED',
     *     STATUS='UNKNOWN')
      OPEN(UNIT=13,FILE=NEW_COMB,  FORM='FORMATTED',  STATUS='UNKNOWN')

      WRITE(11,15) ERROR_OUT

**Format statements
**
10    FORMAT(I3)
15    FORMAT(A)
17    FORMAT(90(' '))
```

```
14      FORMAT(2(A7,F7.2))
18      FORMAT(A25,I5,A15)
19      FORMAT(A4,F10.2)

**

*initializing the 'new' string
      NEW(1:90)=BLANKSTRING(1:90)
      PLINE(1:90)=BLANKSTRING(1:90)

      READ(1,15,END=500)  PSS
25      READ(2,15,END=500)  PES
        DATE2=PES(8:13)

*
* comparing the dates. When date changes set sequence number (count) back to
* zero.
        IF(DATE2.NE.DATE1)THEN
         COUNT=0
         DATE1=DATE2
         EFFORT=.FALSE.
        ENDIF


*reading the sighting number from the sight string
      READ(PSS(5:7),10)  SNUM_S
      If (SNUM_S.EQ.0)  SNUM_S=-1
*       print'(1X,A30,I3)',"SIGHTFILE  SIGHT NUMBER",snum_s

*reading the number of Repeat Occurences  from the effort string
      READ(PES(36:37),'(I2)')  RO
* Must read each chunk of effort data and convert it. P$ is a position shift
* parameter which tells the computer where to find the same field type at
* different positions in the string.

        DO 50 i=0,RO-1
        P$=I*23

C
* this part of the program checks the searching yes/no data field in the effort
* record. If the previous search yes/no was 'no' and the next is 'yes' -> call R
* This had to be added to the program because frquently there were cases where
* team stopped searchin without 'diverting' or 'ENDING LEG'
C
        PREVSYN=SYN
        SYN=PES(57+P$:57+P$)
        IF(SYN.EQ.' ') SYN=PREVSYN

        IF ((PREVSYN.EQ.'1')  .AND.( SYN.EQ.'2').AND.(EFFORT))
     *   THEN
          CALL E(PES,P$)
          IF(PLINE(1:90).EQ.BLANKSTRING(1:90))  THEN
             COUNT=COUNT+1
             WRITE(NEW(1:3),'(I3)')  COUNT
             PLINE=NEW
             EFFORT=.FALSE.
             ELSE IF((PLINE(4:4).EQ.'R'.AND.  PLINE(6:18).EQ.NEW(6:18)
     *)) THEN
```

```
            PLINE=BLANKSTRING
            count=count-1
            EFFORT=.FALSE.
           ELSE
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
            EFFORT=.FALSE.
          ENDIF
          PREVC='E'
         ENDIF

         IF (PREVSYN.EQ.'2' .AND. SYN.EQ.'1' .AND. (.NOT.EFFORT)) THEN
          CALL R(PES,P$)
            IF(PLINE(1:90).EQ.BLANKSTRING(1:90))  THEN
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            PLINE=NEW
            EFFORT=.TRUE.
           ELSE IF((PLINE(4:4).EQ.'E').AND.(PLINE(6:18).EQ.
     * NEW(6:18))) THEN
            PLINE=BLANKSTRING
            COUNT=COUNT-1
            EFFORT=.TRUE.
           ELSE
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
            NEWEFF=.TRUE.
            EFFORT=.TRUE.
           ENDIF
          PREVC='R'
         ENDIF


*checking the 'reason for position field'.
*In this data format a 1=begin leg, a 2=track check, a 3=divert,
* a 4=overschool, a 5=return, a 6=end leg.

         IF((PES(56+P$:56+P$).EQ.'1').AND.(PES(57+P$:57+P$).EQ.'1')
     *.AND. (.NOT.EFFORT)) THEN
          CALL R(PES,P$)
*
* This IF, ELSE-IF, ELSE block tests to see whether the previous line is an 'E'
* (end effort line) with the same time as this 'R' (resume effort). If so don't
* write either of them to the file. This eliminates the paradox of ending and
* resuming effort simultaneously.
*

            IF(PLINE(1:90).EQ.BLANKSTRING(1:90))  THEN
             COUNT=COUNT+1
             WRITE(NEW(1:3),'(I3)') COUNT
```

```
            PLINE=NEW
            EFFORT=.TRUE.
           ELSE IF((PLINE(4:4).EQ.'E').AND.(PLINE(6:18).EQ.
     * NEW(6:18))) THEN
            PLINE=BLANKSTRING
            COUNT=COUNT-1
            EFFORT=.TRUE.
           ELSE
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            CALL LENGTH (PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
          IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
           NEWEFF=.TRUE.
           EFFORT=.TRUE.
           ENDIF
         PREVC='R'
          goto 60

         ELSE IF((PES(56+P$:56+P$).EQ.'1').AND.(PES(57+P$:57+P$)
     *.EQ.'2')) then
         CALL L(PES,P$)
         NEWEFF=.TRUE.
         COUNT=COUNT+1
         WRITE(NEW(1:3),'(I3)') COUNT
         PREVC='L'
          IF(PLINE.NE.BLANKSTRING) THEN
           CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
          ENDIF

*checking the search yes/no
*if there is no begin leg indicator with a search =no indicator, flag an error.

60        CALL P(PES,NEWEFF,OLDP)
*          PRINT*,NEW
         PREVC='P'
         CALL LENGTH(NEW,LENTH)
          IF(LENTH.GT.40)  THEN
          COUNT=COUNT+1
          WRITE(NEW(1:3),'(I3)') COUNT
           IF(PLINE.NE.BLANKSTRING) THEN
            CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
           ENDIF
            PLINE=NEW
          ENDIF

         CALL A(PES,NEWEFF,OLDA)
*          PRINT*,NEW
         PREVC='A'
         CALL LENGTH(NEW,LENTH)
          IF(LENTH.GT.40)  THEN
          COUNT=COUNT+1
```

```
      WRITE(NEW(1:3),'(I3)') COUNT
       IF(PLINE.NE.BLANKSTRING) THEN
        CALL LENGTH(PLINE,LENTH)
        WRITE(UNIT=13,FMT=15,IOSTAT=IERR30) PLINE(1:LENTH)
        IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
       ENDIF
        PLINE=NEW
      ENDIF

     CALL W(PES,NEWEFF,OLDW)
*     PRINT*,NEW
     PREVC='W'
     CALL LENGTH(NEW,LENTH)
      IF(LENTH.GT.40) THEN
      COUNT=COUNT+1
      WRITE(NEW(1:3),'(I3)') COUNT
       IF(PLINE.NE.BLANKSTRING) THEN
        CALL LENGTH(PLINE,LENTH)
        WRITE(UNIT=13,FMT=15,IOSTAT=IERR30) PLINE(1:LENTH)
        IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
       ENDIF
        PLINE=NEW
      ENDIF


     CALL G(PES,NEWEFF,OLDG)
*     PRINT*,NEW
     PREVC='G'
     CALL LENGTH(NEW,LENTH)
      IF(LENTH.GT.40) THEN
      COUNT=COUNT+1
      WRITE(NEW(1:3),'(I3)') COUNT
       IF(PLINE.NE.BLANKSTRING) THEN
        CALL LENGTH(PLINE,LENTH)
        WRITE(UNIT=13,FMT=15,IOSTAT=IERR30) PLINE(1:LENTH)
        IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
       ENDIF
        PLINE=NEW
      ENDIF

    ENDIF

    IF(PES(56+P$:56+P$).EQ.'2') THEN
    CALL POS(PES,P$)
     PREVC='*'
     COUNT=COUNT+1
       WRITE(NEW(1:3),'(I3)') COUNT
     CALL LENGTH(NEW,LENTH)
       IF(PLINE.NE.BLANKSTRING) THEN
        CALL LENGTH(PLINE,LENTH)
        WRITE(UNIT=13,FMT=15,IOSTAT=IERR30) PLINE(1:LENTH)
        IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
       ENDIF
        PLINE=NEW
      ENDIF

    IF(PES(56+P$:56+P$).EQ.'5' .AND. (.NOT.EFFORT)) THEN
     CALL R(PES,P$)
```

```
            IF(PLINE(1:90).EQ.BLANKSTRING(1:90))   THEN
             COUNT=COUNT+1
             WRITE(NEW(1:3),'(I3)')  COUNT
             PLINE=NEW
             EFFORT=.TRUE.
            ELSE IF(PLINE(4:4).EQ.'E'.AND.PLINE(6:18).EQ.NEW(6:18))
     * THEN
             PLINE=BLANKSTRING
             count=count-1
             EFFORT=.TRUE.
            ELSE
             COUNT=COUNT+1
             WRITE(NEW(1:3),'(I3)')  COUNT
             CALL LENGTH(PLINE,LENTH)
             WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)   PLINE(1:LENTH)
           IF(IERR30.NE.0)  PRINT*,'ERROR  IN WRITING  TO'//NEW_COMB
             PLINE=NEW
             NEWEFF=.TRUE.
             EFFORT=.TRUE.
            ENDIF

          PREVC='R'
         ENDIF


        READ(PES(58+P$:60+P$),10)  SNUM_E
*       print'(1X,A30,I3)', "EFFORT SIGHT NUMBER ",SNUM_E
*checking to see if sight # on sight string matches sight # on effort string.
*if they match, use the time recorded in effort data as time of sighting.
*this had to be done because no times were recorded on the sighting strings
          IF (SNUM_S.EQ.SNUM_E)  THEN
           TIME_S=PES(51+P$:55+P$)
*before listing the sighting infomation. Must determine whether or not the
*sighting was on effort. Since the effort data is not a reliable indicator
*use the search yes/no indicator.
          PREVSYN=SYN
          SYN=PSS(208:208)
          IF(SYN.EQ.' ') SYN=PREVSYN

          IF ((PREVSYN.EQ.'1')  .AND.( SYN.EQ.'2').AND.(EFFORT))
     * THEN
          CALL E(PES,P$)
           IF(PLINE(1:90).EQ.BLANKSTRING(1:90))   THEN
             COUNT=COUNT+1
             WRITE(NEW(1:3),'(I3)')  COUNT
             PLINE=NEW
             EFFORT=.FALSE.
           ELSE IF((PLINE(4:4).EQ.'R'.AND.  PLINE(6:18).EQ.NEW(6:18)
     *)) THEN
             PLINE=BLANKSTRING
             count=count-1
             EFFORT=.FALSE.
           ELSE
             COUNT=COUNT+1
             WRITE(NEW(1:3),'(I3)')  COUNT
             CALL LENGTH(PLINE,LENTH)
             WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)   PLINE(1:LENTH)
             IF(IERR30.NE.0)  PRINT*,'ERROR  IN WRITING  TO'//NEW_COMB
```

```
          PLINE=NEW
          EFFORT=.FALSE.
         ENDIF
        PREVC='E'
       ENDIF

      IF(PREVSYN.EQ.'2'  .AND. SYN.EQ.'1' .AND. (.NOT.EFFORT)) THEN
       CALL R(PES,P$)
         IF(PLINE(1:90).EQ.BLANKSTRING(1:90))  THEN
          COUNT=COUNT+1
          WRITE(NEW(1:3),'(I3)') COUNT
          PLINE=NEW
          EFFORT=.TRUE.
         ELSE IF((PLINE(4:4).EQ.'E').AND.(PLINE(6:18).EQ.
     * NEW(6:18))) THEN
          PLINE=BLANKSTRING
          COUNT=COUNT-1
          EFFORT=.TRUE.
         ELSE
          COUNT=COUNT+1
          WRITE(NEW(1:3),'(I3)') COUNT
          CALL LENGTH(PLINE,LENTH)
          WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
          IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
          PLINE=NEW
          NEWEFF=.TRUE.
          EFFORT=.TRUE.
         ENDIF
        PREVC='R'
       ENDIF

       CALL SIGHT(TIME_S)
*       PRINT*,NEW
       COUNT=COUNT+1
         WRITE(NEW(1:3),'(I3)') COUNT
       PREVC='S'
         IF(PLINE.NE.BLANKSTRING)  THEN
          CALL LENGTH(PLINE,LENTH)
          WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
          IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
         ENDIF
          PLINE=NEW

       CALL ONE
*       PRINT*,NEW
       PREVC='#'
       CALL LENGTH(NEW,LENTH)
        IF(LENTH.GT.10)  THEN
         IF(PLINE.NE.BLANKSTRING)  THEN
          CALL LENGTH(PLINE,LENTH)
          WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
          IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
         ENDIF
          PLINE=NEW
        ENDIF

        DO 100 J=2,5
         CALL NUM(J)
```

```
          PREVC='#'
*          PRINT*,NEW
          CALL LENGTH(NEW,LENTH)
          IF(LENTH.GT.10)  THEN
          IF(PLINE.NE.BLANKSTRING)  THEN
           CALL LENGTH(PLINE,LENTH)
           WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
           IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
          ENDIF
           PLINE=NEW
          ENDIF

100        CONTINUE

         CALL WEATHER(TIME_S)
*          PRINT*,NEW
          COUNT=COUNT+1
          WRITE(NEW(1:3),'(I3)')  COUNT
          PREVC='W'
          CALL LENGTH(NEW,LENTH)
          IF(LENTH.GT.40)  THEN
          IF(PLINE.NE.BLANKSTRING)  THEN
           CALL LENGTH(PLINE,LENTH)
           WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
           IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
          ENDIF
           PLINE=NEW
          ENDIF

         CALL GLARE(TIME_S)
*          PRINT*,NEW
          PREVC='G'
          COUNT=COUNT+1
          WRITE(NEW(1:3),'(I3)')  COUNT
          CALL LENGTH(NEW,LENTH)
          IF(LENTH.GT.40)  THEN
          IF(PLINE.NE.BLANKSTRING)  THEN
           CALL LENGTH(PLINE,LENTH)
           WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
           IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
          ENDIF
           PLINE=NEW
          ENDIF


         READ(1,15,END=125)  PSS
         READ(PSS(5:7),10)  SNUM_S
         IF(SNUM_S.EQ.0)  SNUM_S=-1

        PREVSYN=SYN
        SYN=PES(57+P$:57+P$)
        IF(SYN.EQ.'  ') SYN=PREVSYN

        IF ((PREVSYN.EQ.'1')  .AND.( SYN.EQ.'2').AND.(EFFORT))
     *   THEN
          CALL E(PES,P$)
          IF(PLINE(1:90).EQ.BLANKSTRING(1:90))  THEN
            COUNT=COUNT+1
```

```
            WRITE(NEW(1:3),'(I3)') COUNT
            PLINE=NEW
            EFFORT=.FALSE.
            ELSE IF((PLINE(4:4).EQ.'R'.AND. PLINE(6:18).EQ.NEW(6:18)
     *)) THEN
            PLINE=BLANKSTRING
            count=count-1
            EFFORT=.FALSE.
           ELSE
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30) PLINE(1:LENTH)
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
            EFFORT=.FALSE.
          ENDIF
          PREVC='E'
        ENDIF

        IF(PREVSYN.EQ.'2' .AND. SYN.EQ.'1' .AND. (.NOT.EFFORT)) THEN
          CALL R(PES,P$)
            IF(PLINE(1:90).EQ.BLANKSTRING(1:90)) THEN
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            PLINE=NEW
            EFFORT=.TRUE.
           ELSE IF((PLINE(4:4).EQ.'E').AND.(PLINE(6:18).EQ.
     * NEW(6:18))) THEN
            PLINE=BLANKSTRING
            COUNT=COUNT-1
            EFFORT=.TRUE.
           ELSE
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30) PLINE(1:LENTH)
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
            NEWEFF=.TRUE.
            EFFORT=.TRUE.
           ENDIF
         PREVC='R'
         ENDIF

        ENDIF

125      CONTINUE



        IF(EFFORT) THEN

         IF((PES(56+P$:56+P$).EQ.'3').OR.(PES(56+P$:56+P$).EQ.'6'))
     * THEN
          CALL E(PES,P$)

            IF(PLINE(1:90).EQ.BLANKSTRING(1:90)) THEN
```

```
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            PLINE=NEW
            EFFORT=.FALSE.
           ELSE IF(PLINE(4:4).EQ.'R'.AND.PLINE(6:18).EQ.NEW(6:18))
     *      THEN
            PLINE=BLANKSTRING
            count=count-1
            EFFORT=.FALSE.
           ELSE
            COUNT=COUNT+1
            WRITE(NEW(1:3),'(I3)') COUNT
            CALL LENGTH(PLINE,LENTH)
            WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
            IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
            PLINE=NEW
            EFFORT=.FALSE.
          ENDIF

          PREVC='E'
         ENDIF
        ENDIF


        IF(PES(56+P$:56+P$).EQ.'4') THEN
        CALL POS(PES,P$)
         PREVC='*'
        COUNT=COUNT+1
          WRITE(NEW(1:3),'(I3)') COUNT
        CALL LENGTH(NEW,LENTH)
          IF(PLINE.NE.BLANKSTRING) THEN
           CALL LENGTH(PLINE,LENTH)
           WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE(1:LENTH)
           IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB
           PLINE=NEW
          ENDIF

         ENDIF
       NEWEFF=.FALSE.

50      CONTINUE

       goto 25

500     WRITE(UNIT=13,FMT=15,IOSTAT=IERR30)  PLINE
       IF(IERR30.NE.0) PRINT*,'ERROR IN WRITING TO'//NEW_COMB

850     REWIND(13)
851     I=0
       READ(13,15)NEW1
       I=I+1
       READ(NEW1(13:18),15)DATE1
         new1(4:4)=cd1
       if ((cd1.eq.'1') .or. (cd1.eq.'2') .or. (cd1.eq.'3') .or. (cd1
     *.eq.'4') .or. (cd1.eq.'5') .or. (cd1.eq.'6')) goto 851
*reading the minutes & degrees, converting them to degrees & decimal
* degrees
```

```
      READ(NEW1(24:35),'(I2)')MINLAT
      READ(NEW1(35:36),'(I2)')MINLONG
      READ(NEW1(21:22),'(I2)')DGLAT
      READ(NEW1(31:33),'(I3)')DGLONG

      DECLAT=(FLOAT(MINLAT))/60
      DECLONG=(FLOAT(MINLONG))/60
      RLAT1=FLOAT(DGLAT)  + DECLAT
      RLONG1=FLOAT(DGLONG)+  DECLONG
      IF (NEW1(20:20).EQ.'S')  RLAT1= -1*RLAT1
      IF (NEW1(30:30).EQ.'E')RLONG1=  -1*RLONG1

      READ(NEW1(6:7),'(I2)')HR1
      READ(NEW1(8:9),'(I2)')MIN1
      DECTIME1=(FLOAT(MIN1))/60
      TIME1=FLOAT(HR1)+DECTIME1

855    read(UNIT=13,FMT=15,END=800)NEW2
      I=I+1
*checking to see if any position was recorded
       cd2=new2(4:4)
      if ((cd2.eq.'1') .or. (cd2.eq.'2') .or. (cd2.eq.'3') .or. (cd2
   *.eq.'4') .or. (cd2.eq.'5') .or. (cd2.eq.'6')) goto 855
* calculating the time in hrs and decimal hours
      READ(new2(6:7),'(I2)')HR2
      READ(new2(8:9),'(I2)')MIN2
      DECTIME2=(FLOAT(MIN2))/60
      TIME2=FLOAT(HR2)+DECTIME2
* checking for a time change (no time change, no need to calculate dist.)
      IF(TIME2.EQ.TIME1)  GOTO 855

      READ(new2(24:35),'(I2)')MINLAT
      READ(new2(35:36),'(I2)')MINLONG
      READ(new2(21:22),'(I2)')DGLAT
      READ(new2(31:33),'(I3)')DGLONG

      DECLAT=(FLOAT(MINLAT))/60
      DECLONG=(FLOAT(MINLONG))/60
      RLAT2=FLOAT(DGLAT)  + DECLAT
      RLONG2=FLOAT(DGLONG)+  DECLONG
      IF (new2(20:20).EQ.'S')  RLAT2= -1*RLAT2
      IF (new2(30:30).EQ.'E')RLONG2=  -1*RLONG2

*checking the dates. if different days dont calculate speed
      READ(new2(13:18),15)DATE2
      IF (DATE1.NE.DATE2)  THEN
       TIME1=TIME2
       DATE1=DATE2
       RLAT1=RLAT2
       RLONG1=RLONG2
         cd1=cd2
         new1=new2
       GOTO 855
      ENDIF


*using the grcirc subroutine to calculate distance between points
```

```
        CALL GRCIRC(RLAT1,RLONG1,RLAT2,RLONG2,DIST)
       IF (DIST.GT.20) THEN
        VEL=DIST/(TIME2-TIME1)
        DTIME=(TIME2-TIME1)
         IF((VEL.GT.200).OR.(VEL.LT.0))  THEN
          PRINT*,'POSSIBLE  ERROR IN LINE.',I,'  UNLIKELY  VEL. '
          WRITE(11,15)' '
          WRITE(11,18)'POSSIBLE  ERROR IN LINE.',I,'  UNLIKELY  VEL. '
          WRITE(11,19)'VEL=',VEL
            WRITE(11,14)'DTIME=',DTIME

*determining  which file the positions came from
             if((cd1.eq.'P').OR.(cd1.EQ.'R').OR.(cd1.EQ.'A')
     *      .OR.(cd1.EQ.'W').OR.(cd1.EQ.'G').OR.(cd1.EQ.'*'))   THEN
                WRITE(11,15)  'POSITION1  FROM  EFFORT  FILE'
             else
                  write(11,15)'POSITION1   FROM  THE  SIGHTING  FILE'
           endif
             WRITE(11,15)  'TIME,  DATE,  AND  POS  ', new1(6:40)
               write(11,15)  ' '

             if((cd2.eq.'P').OR.(CD2.EQ.'R').OR.(CD2.EQ.'A').OR.
     *      (CD2.EQ.'W').OR.(cd2.EQ.'G').OR.(cd2.EQ.'*'))   THEN
                WRITE(11,15)  'POSITION2  FROM  EFFORT  FILE'
             else
                  write(11,15)'POSITION2   FROM  THE  SIGHTING  FILE'
           endif
             WRITE(11,15)  'TIME,  DATE,  AND  POS  ', new2(6:40)
             write(11,15)  ' '
          ENDIF
        ENDIF
        TIME1=TIME2
        DATE1=DATE2
        RLAT1=RLAT2
        RLONG1=RLONG2
          cd1=cd2
          new1=new2
        GOTO 855


800         END
*****************************************************************************
        SUBROUTINE  E(PES,P$)
*
*
         CHARACTER  PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
         CHARACTER  TIME_E*5
         COMMON  NEW,B$,PSS,BLANKSTRING

*Initializing the string
        NEW(1:90)=BLANKSTRING(1:90)
*Event code
        NEW(4:4)='E'

*on or off effort
        IF(PES(57+P$:57+P$).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
```

```
          TIME_E=PES(51+P$:55+P$)
          READ(TIME_E(5:5),'(F2.1)')  DECMIN
          ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.'  ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43+p$:43+p$).eq.'1')  NEW(20:20)='N'
      If(pes(43+p$:43+p$).eq.'2')  NEW(20:20)='S'
      If(pes(50+p$:50+p$).eq.'1')  NEW(30:30)='E'
      If(pes(50+p$:50+p$).eq.'2')  NEW(30:30)='W'

      NEW(21:28)=PES(38+P$:39+P$)//':'//PES(40+P$:41+P$)//'.'//
     *  PES(42+P$:42+P$)//'  '

      NEW(31:39)=PES(44+P$:46+P$)//':'//PES(47+P$:48+P$)//'.'//
     *  PES(49+P$:49+P$)//'  '

      RETURN
      END
*************************************************************************
      SUBROUTINE  R(PES,P$)
*
*
       CHARACTER  PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
       CHARACTER  TIME_E*5
       COMMON  NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='R'

*on or off effort
      IF(PES(57+P$:57+P$).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
          TIME_E=PES(51+P$:55+P$)
          READ(TIME_E(5:5),'(F2.1)')  DECMIN
          ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.'  ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43+p$:43+p$).eq.'1')  NEW(20:20)='N'
      If(pes(43+p$:43+p$).eq.'2')  NEW(20:20)='S'
      If(pes(50+p$:50+p$).eq.'1')  NEW(30:30)='E'
      If(pes(50+p$:50+p$).eq.'2')  NEW(30:30)='W'
```

```
      NEW(21:28)=PES(38+P$:39+P$)//':'//PES(40+P$:41+P$)//'.'//
     *  PES(42+P$:42+P$)//'  '

      NEW(31:39)=PES(44+P$:46+P$)//':'//PES(47+P$:48+P$)//'.'//
     *  PES(49+P$:49+P$)//'  '

      RETURN
      END
*************************************************************************
      SUBROUTINE L(PES,P$)
*
*
       CHARACTER PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
       CHARACTER TIME_E*5
       COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='L'

*on or off effort
      IF(PES(57+P$:57+P$).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
          TIME_E=PES(51+P$:55+P$)
          READ(TIME_E(5:5),'(F2.1)')  DECMIN
          ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.'  ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43+p$:43+p$).eq.'1')  NEW(20:20)='N'
      If(pes(43+p$:43+p$).eq.'2')  NEW(20:20)='S'
      If(pes(50+p$:50+p$).eq.'1')  NEW(30:30)='E'
      If(pes(50+p$:50+p$).eq.'2')  NEW(30:30)='W'

      NEW(21:28)=PES(38+P$:39+P$)//':'//PES(40+P$:41+P$)//'.'//
     *  PES(42+P$:42+P$)//'  '

      NEW(31:39)=PES(44+P$:46+P$)//':'//PES(47+P$:48+P$)//'.'//
     *  PES(49+P$:49+P$)//'  '

      NEW(43:44)=PES(3:4)


      RETURN
      END

*************************************************************************
      SUBROUTINE W(PES,NEWEFF,OLDW)
*
*this subroutine creates a 'W' event
```

```
*
      CHARACTER PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
      CHARACTER TIME_E*5,OLDW*90
      LOGICAL NEWEFF
      COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='W'

*on or off effort
      IF(PES(57:57).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
         TIME_E=PES(51:55)
         READ(TIME_E(5:5),'(F2.1)') DECMIN
         ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.' ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43:43).eq.'1') NEW(20:20)='N'
      If(pes(43:43).eq.'2') NEW(20:20)='S'
      If(pes(50:50).eq.'1') NEW(30:30)='E'
      If(pes(50:50).eq.'2') NEW(30:30)='W'

      NEW(21:28)=PES(38:39)//':'//PES(40:41)//'.'//
     *  PES(42:42)//' '

      NEW(31:39)=PES(44:46)//':'//PES(47:48)//'.'//
     *  PES(49:49)//' '

*beaufort
      NEW(51:54)='    '//PES(35:35)

      IF (((NEW(41:80) .NE. OLDW(41:80)).OR.(NEWEFF))) THEN
        OLDW(41:80)= NEW(41:80)
      ELSE
        NEW= NEW(1:40)
      ENDIF

      RETURN
      END
*****************************************************************************
      SUBROUTINE G(PES,NEWEFF,OLDG)
*
*this subroutine creates a 'G' event
*
      CHARACTER PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
      CHARACTER TIME_E*5,OLDG*90
      LOGICAL NEWEFF
      COMMON NEW,B$,PSS,BLANKSTRING
```

```
*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='G'

*on or off effort
      IF(PES(57:57).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
         TIME_E=PES(51:55)
         READ(TIME_E(5:5),'(F2.1)')  DECMIN
         ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.' ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43:43).eq.'1')  NEW(20:20)='N'
      If(pes(43:43).eq.'2')  NEW(20:20)='S'
      If(pes(50:50).eq.'1')  NEW(30:30)='E'
      If(pes(50:50).eq.'2')  NEW(30:30)='W'

      NEW(21:28)=PES(38:39)//':'//PES(40:41)//'.'//
     *  PES(42:42)//' '

      NEW(31:39)=PES(44:46)//':'//PES(47:48)//'.'//
     *  PES(49:49)//' '

*horizontal sun position
      NEW(43:44)=PES(31:32)

*vertical sun position
      NEW(48:49)=PES(33:34)

      IF (((NEW(41:80) .NE. OLDG(41:80)).OR.(NEWEFF)))  THEN
        OLDG(41:80)= NEW(41:80)
      ELSE
        NEW= NEW(1:40)
      ENDIF


      RETURN
      END

*************************************************************************
      SUBROUTINE  A(PES,NEWEFF,OLDA)
*
*this subroutine creates an 'A' event
*
       CHARACTER  PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
       CHARACTER  TIME_E*5,OLDA*90
       LOGICAL  NEWEFF
       COMMON  NEW,B$,PSS,BLANKSTRING
```

```
*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='A'

*on or off effort
      IF(PES(57:57).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
         TIME_E=PES(51:55)
         READ(TIME_E(5:5),'(F2.1)')  DECMIN
         ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.' ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43:43).eq.'1')  NEW(20:20)='N'
      If(pes(43:43).eq.'2')  NEW(20:20)='S'
      If(pes(50:50).eq.'1')  NEW(30:30)='E'
      If(pes(50:50).eq.'2')  NEW(30:30)='W'

      NEW(21:28)=PES(38:39)//':'//PES(40:41)//'.'//
     *  PES(42:42)//' '

      NEW(31:39)=PES(44:46)//':'//PES(47:48)//'.'//
     *  PES(49:49)//' '

*altitude
      NEW(41:44)=PES(14:17)
*speed
      NEW(46:49)=' '//PES(18:20)

      IF (((NEW(41:80) .NE. OLDA(41:80)).OR.(NEWEFF)))  THEN
        OLDA(41:80)= NEW(41:80)
      ELSE
        NEW= NEW(1:40)
      ENDIF


      RETURN
      END

*****************************************************************************
      SUBROUTINE  P(PES,NEWEFF,OLDP)
*
*
       CHARACTER  PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
       CHARACTER  TIME_E*5,OLDP*90
       LOGICAL  NEWEFF
       COMMON  NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
```

```
*Event code
      NEW(4:4)='P'

*on or off effort
      IF(PES(57:57).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
         TIME_E=PES(51:55)
         READ(TIME_E(5:5),'(F2.1)')  DECMIN
         ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.'  ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43:43).eq.'1')  NEW(20:20)='N'
      If(pes(43:43).eq.'2')  NEW(20:20)='S'
      If(pes(50:50).eq.'1')  NEW(30:30)='E'
      If(pes(50:50).eq.'2')  NEW(30:30)='W'

      NEW(21:28)=PES(38:39)//':'//PES(40:41)//'.'//
     *  PES(42:42)//'  '

      NEW(31:39)=PES(44:46)//':'//PES(47:48)//'.'//
     *  PES(49:49)//'  '

*lt obs
      NEW(41:44)=B$//PES(25:26)
*bow obs
      NEW(46:49)=B$//PES(21:22)
*rt obs
      NEW(51:54)=B$//PES(27:28)

*field 4: off effort observer (not recorded)
*field 5: off effort observer (not recorded)

*recorder
      NEW(66:69)=B$//PES(29:30)

*checking for duplication. if no change, dont write it to file.
      IF (((NEW(41:80) .NE. OLDP(41:80)).OR.(NEWEFF)))  THEN
        OLDP(41:80)= NEW(41:80)
      ELSE
        NEW= NEW(1:40)
      ENDIF


      RETURN
      END
*************************************************************************
      SUBROUTINE  POS(PES,P$)
*
*
       CHARACTER  PES*500,NEW*90,PSS*250,B$*2,BLANKSTRING*90
```

```
      CHARACTER TIME_E*5
      COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='*'

*on or off effort
      IF(PES(57+P$:57+P$).EQ.'1')NEW(5:5)='.'

*Converting the decimal minutes into seconds
         TIME_E=PES(51+P$:55+P$)
         READ(TIME_E(5:5),'(F2.1)') DECMIN
         ISEC=DECMIN*60
*Time
      NEW(6:9)=TIME_E(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.' ') NEW(10:10)='0'

*date
      NEW(13:18)=PES(10:13)//PES(8:9)

*position
      If(pes(43+p$:43+p$).eq.'1') NEW(20:20)='N'
      If(pes(43+p$:43+p$).eq.'2') NEW(20:20)='S'
      If(pes(50+p$:50+p$).eq.'1') NEW(30:30)='E'
      If(pes(50+p$:50+p$).eq.'2') NEW(30:30)='W'

      NEW(21:28)=PES(38+P$:39+P$)//':'//PES(40+P$:41+P$)//'.'//
     *  PES(42+P$:42+P$)//'  '

      NEW(31:39)=PES(44+P$:46+P$)//':'//PES(47+P$:48+P$)//'.'//
     *  PES(49+P$:49+P$)//'  '

      RETURN
      END
*************************************************************************
      SUBROUTINE SIGHT (TIME_S)
*
*This subroutine creates a sighting event. code 's'
*
      CHARACTER TIME_S*5,NEW*90,PSS*250,B$*2,BLANKSTRING*90
      COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Converting the decimal minutes into seconds
      READ(TIME_S(5:5),'(F2.1)') DECMIN
      ISEC=DECMIN*60
*Event code
      NEW(4:4)='S'
*On or off effort
       IF(PSS(208:208).EQ.'1') NEW(5:5)='.'
*Time
      NEW(6:9)=TIME_S(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.' ') NEW(10:10)='0'
```

```
*date
      NEW(13:18)=PSS(11:14)//PSS(9:10)
*Position
      IF (PSS(41:41).EQ.'1')  NEW(20:20)='N'
      IF (PSS(41:41).EQ.'2')  NEW(20:20)='S'
      IF (PSS(47:47).EQ.'1')  NEW(30:30)='E'
      IF (PSS(47:47).EQ.'2')  NEW(30:30)='W'

      NEW(21:28)=PSS(37:38)//':'//PSS(39:40)//'.'//B$
      NEW(31:39)=PSS(42:44)//':'//PSS(45:46)//'.'//B$

*Sight number
      IF((PSS(5:7).NE.'   ').AND.(PSS(6:6).EQ.' '))
     *   PSS(6:6)='0'
      NEW(41:44)='  '//PSS(5:7)
*Obs number
      NEW(46:49)=B$//PSS(31:32)
*Declination angle is not recorded
*For simplicity Perpendicular distance has been placed in the data field
* that declination angle should occupy.
      NEW(51:54)=PSS(33:34)//'.'//PSS(35:35)

*Primary observer y/n? all observers considered primary
      NEW(56:59)='   '//'Y'
*Spp 1 code
      IF((PSS(65:66).NE.'  ').AND.(PSS(65:65).EQ.' '))
     *   PSS(65:65)='0'
      NEW(61:64)=B$//PSS(65:66)
*Spp 2 code
      IF((PSS(99:100).NE.'  ').AND.(PSS(99:99).EQ.' '))
     *   PSS(99:99)='0'
      NEW(66:69)=B$//PSS(99:100)
*Spp 3 code
      IF((PSS(118:119).NE.'  ').AND.(PSS(118:118).EQ.' '))
     *   PSS(118:118)='0'
      NEW(71:74)=B$//PSS(118:119)
*Spp 4 code
      IF((PSS(137:138).NE.'  ').AND.(PSS(137:137).EQ.' '))
     *   PSS(137:137)='0'
      NEW(76:79)=B$//PSS(137:138)
*Spp 5 code
      IF((PSS(156:157).NE.'  ').AND.(PSS(156:156).EQ.' '))
     *   PSS(156:156)='0'
      NEW(81:84)=B$//PSS(156:157)
*Perpendicular distance
*       NEW(86:90)=PSS(33:34)//'.'//PSS(35:35)
*       PRINT*,NEW(86:90)
      RETURN
      END
*************************************************************************
      SUBROUTINE GLARE (TIME_S)
*
*This subroutine creates a sun event. code 'g'
*
      CHARACTER TIME_S*5,NEW*90,PSS*250,B$*2,BLANKSTRING*90
      COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
```

```
      NEW(1:90)=BLANKSTRING(1:90)
*Converting the decimal minutes into seconds
      READ(TIME_S(5:5),'(F2.1)') DECMIN
      ISEC=DECMIN*60
*Event code
      NEW(4:4)='G'
*On or off effort
       IF(PSS(208:208).EQ.'1') NEW(5:5)='.'
*Time
      NEW(6:9)=TIME_S(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.' ') NEW(10:10)='0'
*date
      NEW(13:18)=PSS(11:14)//PSS(9:10)
*Position
      IF (PSS(41:41).EQ.'1') NEW(20:20)='N'
      IF (PSS(41:41).EQ.'2') NEW(20:20)='S'
      IF (PSS(47:47).EQ.'1') NEW(30:30)='E'
      IF (PSS(47:47).EQ.'2') NEW(30:30)='W'

      NEW(21:28)=PSS(37:38)//':'//PSS(39:40)//'.'//B$
      NEW(31:39)=PSS(42:44)//':'//PSS(45:46)//'.'//B$

*horizontal sun
      NEW(43:44)=PSS(23:24)

*vertical sun
      NEW(48:49)=PSS(25:26)
* field 3, perpendicular distance, was moved to the sighting line to
* facilitate easy processing of the data.
*  perpendicular distance
*       NEW(51:54)=PSS(33:34)//'.'//PSS(35:35)

*distance determined using what method
      NEW(59:59)=PSS(36:36)

*birds(y/n)
      IF(PSS(174:174).EQ.'1') NEW(64:64)='Y'
      IF(PSS(174:174).EQ.'2') NEW(64:64)='N'

*sea surface temperature
     IF(PSS(27:29).NE.'   ') NEW(67:69)=PSS(27:28)//'.'//PSS(29:29)

      RETURN
      END
*************************************************************************
      SUBROUTINE WEATHER (TIME_S)
*
*This subroutine creates a weather event. code 'W'
*
      CHARACTER TIME_S*5,NEW*90,PSS*250,B$*2,BLANKSTRING*90
      COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
      NEW(1:80)=BLANKSTRING(1:80)
*Converting the decimal minutes into seconds
      READ(TIME_S(5:5),'(F2.1)') DECMIN
      ISEC=DECMIN*60
```

```
*Event code
      NEW(4:4)='W'
*On or off effort
       IF(PSS(208:208).EQ.'1')  NEW(5:5)='.'
*Time
      NEW(6:9)=TIME_S(1:4)
      WRITE(NEW(10:11),'(I2)')ISEC
      IF(NEW(10:10).EQ.'  ') NEW(10:10)='0'
*date
      NEW(13:18)=PSS(11:14)//PSS(9:10)
*Position
      IF (PSS(41:41).EQ.'1')  NEW(20:20)='N'
      IF (PSS(41:41).EQ.'2')  NEW(20:20)='S'
      IF (PSS(47:47).EQ.'1')  NEW(30:30)='E'
      IF (PSS(47:47).EQ.'2')  NEW(30:30)='W'

      NEW(21:28)=PSS(37:38)//':'//PSS(39:40)//'.'//B$
      NEW(31:39)=PSS(42:44)//':'//PSS(45:46)//'.'//B$
*Beaufort sea state
            NEW(51:54)='    '//pss(30:30)

      return
      end

*************************************************************************
      SUBROUTINE ONE
*
*This subroutine creates a ONE event. codes 1
*
      CHARACTER NEW*90,PSS*250,B$*2,BLANKSTRING*90
      COMMON NEW,B$,PSS,BLANKSTRING
      INTEGER P$

*Initializing the string
      NEW(1:90)=BLANKSTRING(1:90)
*Event code
      NEW(4:4)='1'
*On or off effort
       IF(PSS(208:208).EQ.'1')  NEW(5:5)='.'

*observer code
      NEW(41:44)=b$//PSS(48:49)

*best estimate of school size
      NEW(46:49)=PSS(50:53)

*high estimate of school size
      NEW(51:54)=PSS(54:57)

*low estimate of school size
      NEW(56:59)=PSS(58:61)

*%SPP1
      NEW(61:64)='  '//PSS(62:64)
*%SPP I
      DO 50 I=2,5
      P$=(I-2)*19
      NEW(66+5*(I-2):69+5*(I-2))='   '//PSS(96+p$:98+p$)
```

```
50      CONTINUE

        RETURN
        END
*****************************************************************************
        SUBROUTINE NUM(I)
*
*This subroutine creates a NUMBER event. codes 2-6
*
        CHARACTER NEW*90,PSS*250,B$*2,BLANKSTRING*90
        COMMON NEW,B$,PSS,BLANKSTRING

*Initializing the string
        NEW(1:90)=BLANKSTRING(1:90)


          P$=(I-2)*19
          IF(PSS(82+P$:83+P$).NE.'    ') THEN
*event code
           WRITE(NEW(4:4),'(I1)')I
*on or off effort
           IF(PSS(208:208).EQ.'1') NEW(5:5)='.'
*observer code
           NEW(41:44)=b$//PSS(82+p$:83+p$)

*best estimate of school size
           NEW(46:49)=PSS(84+p$:87+p$)

*high estimate of school size
           NEW(51:54)=PSS(88+p$:91+p$)

*low estimate of school size
           NEW(56:59)=PSS(92+p$:95+p$)

          ENDIF
10        CONTINUE
        RETURN
        END
*****************************************************************************
        SUBROUTINE LENGTH(NEW,LENTH)
*
* This subroutine estimates the length of a character string excluding
*     the trailing blanks
        CHARACTER*90 NEW
         LENTH= 80
         DO 10 I=90,1,-1
          IF(NEW(I:I).EQ.'  ')THEN
           LENTH=I-1
          ELSE
           RETURN
          ENDIF
10       CONTINUE
        RETURN
        END

*****************************************************************************
      SUBROUTINE GRCIRC(RLAT1,RLONG1,RLAT2,RLONG2,DIST)
C
```

```
C...THIS PROGRAM COMPUTES GREAT CIRCLE DISTANCES BETWEEN TWO COORDINATES
C       USING ALGORITHM TAKEN FROM KEN WALLACE'S PROGRAM (REPUTED TO HAVE
C       BEEN LIFTED FROM H.P. PROGRAM MANUAL).  COORDINATES ARE EXPRESSED
C       AS DEGREES, AND ARE ASSUMED TO BE IN NORTHERN HEMISPHERE, WEST
C       LONGITUDE IF POSITIVE.
C
      IMPLICIT DOUBLE PRECISION (B-H, O-Z)
      DOUBLE PRECISION LAS,LAC,LAF
      real*4 RLAT1,RLONG1,RLAT2,RLONG2,DIST
      DACOS(X)= (3.1415926535 / 2.0) - DATAN( X / DSQRT(1.0 - X*X) )
      DIST= 0.0
      PI= 3.141592653589793
      R= PI / 180.0
      D= 180.0 / PI
C
C...COMPUTE GREAT CIRCLE DISTANCE
      LAS= (DSIN(RLAT1*R)) * (DSIN(RLAT2*R))
      LAC= (DCOS(RLAT1*R)) * (DCOS(RLAT2*R)) *
     1     (DCOS((RLONG1 - RLONG2)*R))
      LAF= LAS + LAC
      IF (LAF .GT. 1.0) LAF=  1.0
      IF (LAF .LT.-1.0) LAF= -1.0
      IF (LAF .EQ. 1.0) GO TO 999
      DIST= (DACOS(LAF)) * D * 60.0
  999 CONTINUE
      RETURN
      END
```

****

Format of **AE74DB** version 1

1 of 2

RECORD NAME	FROM		TO	WIDTH
AE74REC	**1**	**-**	**80**	**80**

ELEMENT NAME	FROM		TO	WIDTH
FLIGHT-AND-LEG -	**1**	**-**	**4**	**4**
- FLIGHT	1	-	2	2
- LEG	3	-	4	2
DATE -	**5**	**-**	**10**	**6**
- MONTH	5	-	6	2
- DAY	7	-	8	2
- YEAR	9	-	10	2
TIME-START-LEG -	**11**	**-**	**14**	**4**
- HR-START-LEG	11	-	12	2
- MIN-START-LEG	13	-	14	2
POSITION-START -	**15**	**-**	**27**	**13**
- LAT-START -	15	-	19	5
- LATD-START	15	-	16	2
- LATM-START	17	-	19	3
- N-OR-S-START	20	-	20	1
- LONG-START -	21	-	26	6
- LONGD-START	21	-	23	3
- LONGM-START	24	-	26	3
- E-OR-W-START	27	-	27	1
TRUE-COURSE	**28**	**-**	**30**	**3**
TIME-END-LEG -	**31**	**-**	**34**	**4**
- HR-END-LEG	31	-	32	2
- MIN-END-LEG	33	-	34	2
POSITION-END -	**35**	**-**	**47**	**13**
- LAT-END -	35	-	39	5
- LATD-END	35	-	36	2
- LATM-END	37	-	39	3
- N-OR-S-END	40	-	40	1
- LONG-END -	41	-	46	6
- LONGD-END	41	-	43	3
- LONGM-END	44	-	46	3
- E-OR-W-END	47	-	47	1
ALTITUDE	**48**	**-**	**51**	**4**
GROUND-SPEED	**52**	**-**	**54**	**3**
VISIBILITY	**55**	**-**	**56**	**2**
BEAUFORT	**57**	**-**	**57**	**1**
LEFT-PATH-WIDTH	**58**	**-**	**59**	**2**
RIGHT-PATH-WIDTH	**60**	**-**	**61**	**2**
TOTAL-SIGHTINGS	**62**	**-**	**63**	**2**
TOTAL-SCHOOLS	**64**	**-**	**65**	**2**
TOTAL-TUNABOATS	**66**	**-**	**67**	**2**
TOTAL-OTHERS	**68**	**-**	**69**	**2**
TIMEOUT-MINUTES	**70**	**-**	**72**	**3**
DIRECTION-GLIT	**73**	**-**	**75**	**3**

Appendix 3

Format of AE74DB version 1

2 of 2

ELEMENT NAME	FROM		TO	WIDTH
PERCENT-GLIT	76	-	77	2
FILLER	78	-	80	3

Appendix 3

Format of AS74DB version 1

1 of 2

RECORD NAME	FROM		TO	WIDTH
AS74REC	1	-	135	135

ELEMENT NAME	FROM		TO	WIDTH
FLIGHT-LEG-SIGHT	1	-	6	6
FLIGHT-LEG-SITE	1	-	6	6
FLIGHT	7	-	8	2
LEG	9	-	10	2
SIGHT	11	-	12	2
DATE -	13	-	18	6
- MONTH	13	-	14	2
- DAY	15	-	16	2
YEAR	17	-	18	2
TIME-OF-SIGHTING -	19	-	22	4
- HR-SIGHT	19	-	20	2
- MIN-SIGHT	21	-	22	2
POSITION -	23	-	35	13
- LATITUDE -	23	-	27	5
- LATD	23	-	24	2
- LATM	25	-	27	3
- N-OR-S	28	-	28	1
- LONGITUDE -	29	-	34	6
- LONGD	29	-	31	3
- LONGM	32	-	34	3
- E-OR-W	35	-	35	1
OBSERVER-NUM	36	-	37	2
FILLER	38	-	38	1
SIGHTING-AREA	39	-	39	1
DISTANCE	40	-	42	3
ANGLE	43	-	45	3
STRUCTURE	46	-	46	1
BEHAVIOR	47	-	47	1
DIRECTION	48	-	50	3
AIDS	51	-	51	1
TOTAL-IN-SCHOOL	52	-	56	5
ERROR-EST-SCHOOL	57	-	60	4
SIGHTING-CODE	61	-	62	2
SP1-SCHOOL	63	-	66	4
SP1-EST	67	-	70	4
SP1-CODE	71	-	72	2
SP2-SCHOOL	73	-	76	4
SP2-EST	77	-	80	4
SP2-CODE	81	-	82	2
SP3-SCHOOL	83	-	86	4
FILLER	87	-	87	1
SP3-EST	88	-	91	4
SP3-CODE	92	-	93	2
QUALITY-CODE	94	-	94	1

Appendix 3

Format of AS74DB version 1

2 of 2

ELEMENT NAME	FROM		TO	WIDTH
PERCENT-UW	**95**	-	**96**	**2**
FISH	**97**	-	**97**	**1**
TONS-FISH	**98**	-	**100**	**3**
FO-TIME	**101**	-	**102**	**2**
FO-RESPONSE	**103**	-	**103**	**1**
FO-ALTITUDE	**104**	-	**107**	**4**
FO-POSITION -	**108**	-	**120**	**13**
- FO-LATITUDE -	108	-	112	5
- FO-LATD	108	-	109	2
- FO-LATM	110	-	112	3
- FO-N-OR-S	113	-	113	1
- FO-LONGITUDE -	114	-	119	6
- FO-LONGD	114	-	116	3
- FO-LONGM	117	-	119	3
- FO-E-OR-W	120	-	120	1
NUM-TUNABOATS	**121**	-	**122**	**2**
FILLER	**123**	-	**135**	**13**

Appendix 4

Format of ASyrDB version 2

1 of 2

RECORD NAME	FROM		TO	WIDTH
AS(77,79,81-82)REC	1	-	241	241

ELEMENT NAME	FROM		TO	WIDTH
FLIGHT-LEG-SIGHT	1	-	7	7
FLIGHT-LEG-SITE	1	-	7	7
FLIGHT	8	-	9	2
LEG	10	-	11	2
SIGHT	12	-	14	3
FILLER	15	-	15	1
DATE -	16	-	21	6
- YEAR	16	-	17	2
- MONTH	18	-	19	2
- DAY	20	-	21	2
BOW-OBS-CODE	22	-	23	2
LEFT-OBS-CODE	24	-	25	2
RIGHT-OBS-CODE	26	-	27	2
RECORDER-CODE	28	-	29	2
HORIZONTAL-SUN	30	-	31	2
VERTICAL-SUN	32	-	33	2
SEA-SURFACE-TEMP	34	-	36	3
BEAUFORT	37	-	37	1
SIGHTER	38	-	39	2
PERPENDICULAR	40	-	42	3
DETERMINED-HOW	43	-	43	1
POSITION -	44	-	54	11
- LATITUDE -	44	-	47	4
- LATD	44	-	45	2
- LATM	46	-	47	2
- N-OR-S	48	-	48	1
- LONGITUDE -	49	-	53	5
- LONGD	49	-	51	3
- LONGM	52	-	53	2
- E-OR-W	54	-	54	1
E1-OBS-CODE	55	-	56	2
E1-BEST-EST	57	-	60	4
E1-HIGHEST-EST	61	-	64	4
E1-LOWEST-EST	65	-	68	4
SP1-PERCENT-TOT	69	-	71	3
SP1-CODE	72	-	73	2
FILLER	74	-	88	15
E2-OBS-CODE	89	-	90	2
E2-BEST-EST	91	-	94	4
E2-HIGHEST-EST	95	-	98	4
E2-LOWEST-EST	99	-	102	4
SP2-PERCENT-TOT	103	-	105	3
SP2-CODE	106	-	107	2
E3-OBS-CODE	108	-	109	2

Appendix 4

Format of ASyrDB version 2

2 of 2

ELEMENT NAME	FROM		TO	WIDTH
E3-BEST-EST	110	-	113	4
E3-HIGHEST-EST	114	-	117	4
E3-LOWEST-EST	118	-	121	4
SP3-PERCENT-TOT	122	-	124	3
SP3-CODE	125	-	126	2
E4-OBS-CODE	127	-	128	2
E4-BEST-EST	129	-	132	4
E4-HIGHEST-EST	133	-	136	4
E4-LOWEST-EST	137	-	140	4
SP4-PERCENT-TOT	141	-	143	3
SP4-CODE	144	-	145	2
E5-OBS-CODE	146	-	147	2
E5-BEST-EST	148	-	151	4
E5-HIGHEST-EST	152	-	155	4
E5-LOWEST-EST	156	-	159	4
SP5-PERCENT-TOT	160	-	162	3
SP5-CODE	163	-	164	2
FILLER	165	-	165	1
E6-OBS-CODE	166	-	167	2
E6-BEST-EST	168	-	171	4
E6-HIGHEST-EST	172	-	175	4
E6-LOWEST-EST	176	-	179	4
PORPOISE	180	-	180	1
BIRDS	181	-	181	1
TUNA	182	-	182	1
INCH9-FLYOVER	183	-	186	4
SCHOOL-MODE	187	-	187	1
ROLL-NUM-70M1	188	-	190	3
FIRST-FRAME-70M1	191	-	193	3
LAST-FRAME-70M1	194	-	196	3
ROLL-NUM-9-IN	197	-	199	3
FIRST-FRAME-9-IN	200	-	202	3
LAST-FRAME-9-IN	203	-	205	3
ROLL-NUM-35M	206	-	208	3
FIRST-FRAME-35M	209	-	211	3
LAST-FRAME-35M	212	-	214	3
GOOD-BAD-UGLY	215	-	215	1
ROLL-NUM-70M2	216	-	218	3
FIRST-FRAME-70M2	219	-	221	3
LAST-FRAME-70M2	222	-	224	3
FILLER	225	-	241	17

Appendix 4

Format of AEyrDB version 2

1 of 2

RECORD NAME	FROM		TO	WIDTH
AE(77,79,81-82)REC	**1**	**-**	**60**	**60**

ELEMENT NAME	FROM		TO	WIDTH
FLIGHT-AND-LEG -	**1**	**-**	**4**	**4**
- FLIGHT	1	-	2	2
- LEG	3	-	4	2
FILLER	**5**	**-**	**7**	**3**
DATE -	**8**	**-**	**13**	**6**
- YEAR	8	-	9	2
- MONTH	10	-	11	2
- DAY	12	-	13	2
ALTITUDE	**14**	**-**	**17**	**4**
GROUND-SPEED	**18**	**-**	**20**	**3**
BOW-OBS-CODE	**21**	**-**	**22**	**2**
FILLER	**23**	**-**	**24**	**2**
LEFT-OBS-CODE	**25**	**-**	**26**	**2**
RIGHT-OBS-CODE	**27**	**-**	**28**	**2**
RECORDER-CODE	**29**	**-**	**30**	**2**
HORIZONTAL-SUN	**31**	**-**	**32**	**2**
VERTICAL-SUN	**33**	**-**	**34**	**2**
BEAUFORT	**35**	**-**	**35**	**1**
REPEAT-OCCURRENCES	**36**	**-**	**37**	**2**
POSITION -	**38**	**-**	**50**	**13**
- LATITUDE -	38	-	42	5
- LATD	38	-	39	2
- LATM	40	-	42	3
- N-OR-S	43	-	43	1
- LONGITUDE -	44	-	49	6
- LONGD	44	-	46	3
- LONGM	47	-	49	3
- E-OR-W	50	-	50	1
GREENWICH-TIME -	**51**	**-**	**55**	**5**
- HR-GMT	51	-	52	2
- MIN-GMT	53	-	55	3
REASON-POSITION	**56**	**-**	**56**	**1**
SEARCHING	**57**	**-**	**57**	**1**
SIGHTING-NUMBER	**58**	**-**	**60**	**3**

THE FOLLOWING GROUP OF ELEMENTS REPEAT UP TO 30 TIMES :
(TOTAL # OF REPEAT GROUPS = # OF 'REPEAT OCCURRANCES')
(REFERENCE ELEMENT AT POSITION 36, SIZE 2.)

ELEMENT NAME	WIDTH
POSITION -	13
- LATITUDE -	5
- LATD	2
- LATM	3
- N-OR-S	1
- LONGITUDE -	6
- LONGD	3
- LONGM	3
- E-OR-W	1
GREENWICH-TIME -	5
- HR-GMT	2
- MIN-GMT	3
REASON-POSITION	1
SEARCHING	1
SIGHTING-NUMBER	3

Southwest Fisheries Science Center, NMFS
Coastal Cetacean Surveys

Key to aerial survey raw data format

Columns	Entry
1-3	Line #
4	Entry type code
5	blank
6-11	Time, format: HHMMSS
12	blank
13-18	Date, format: MMDDYY
19	blank
20-28	Latitude, format: NDD:MM.MM
29	blank
30-39	Longitude, format: WDD:MM.MM
40	blank
41-end	entered data

Key to entry type codes:

* = position and time update (taken every minute)
T = Transect start, automatically followed by V,P,A,W.
V = Viewing conditions for all 3 observer positions. Recorded as percentage of viewing area obscured by glare.
P = Observer positions (order: left front, belly, right front, recorder).
A = Altitude and speed information.
W = Weather information:
- H/K/N — Haze or kelp presence
- 0-100 — % cloud cover
- 0-5 — Beaufort sea state
- G/L/D/R — Water color (green, light blue, dark blue, red tide)

O = End of transect
E = Ending effort on transect (i.e. to circle, transit, ...).
R = Resuming effort.
S = Sighting information:
- Sighting #
- Observer initials
- Declination angle (left side recorded as negative, right, positive)
- Primary sighting? (Y/N) (Note: Belly observer sightings are secondary, because only sightings missed by primary (side) observers may be called by this observer).
- 1-3 fields with IDs for species in sightings

1-6 Individual observer estimates of school size and species proportions. Order: Best, High, Low, SP1 %, SP2%, SP3%
C = Comments - can be entered in any format to add information, point out errors in previous entries, etc.

C O D E T A B L E 4 b
Research Vessel / Life History
Cetacean Species Codes

- - numeric - -

SPEC CODE	SCIENTIFIC NAME	STANDARD COMMON NAME, OTHER COMMON NAMES
01	Mesoplodon peruvianus	Pygmy beaked whale
02	Stenella attenuata subsp.A	Offshore pantropical spotted dolphin, offshore spotter
03	Stenella longirostris subsp.?	Unidentified spinner dolphin, spinner porpoise
04	Stenella clymene	Clymene dolphin, short-snouted spinner dolphin
05	Delphinus delphis subsp.?	Unidentified common dolphin, saddleback dolphin, whitebelly porpoise
06	Stenella attenuata graffmani	Coastal spotted dolphin, spotter, silverbacks
07	Sotalia fluviatilis	Tucuxi, Guiana dolphin
08	Orcaella brevirostris	Irrawaddy dolphin, Lumbalumba
09	Australophocaena dioptrica	Spectacled porpoise
10	S. longirostris orientalis	Eastern spinner dolphin
11	Stenella longirostris hybrid	Whitebelly spinner dolphin
12	Lagenorhynchus albirostris	White-beaked dolphin
13	Stenella coeruleoalba	Striped dolphin, streaker porpoise, euphrosyne dolphin
14	Lagenorhynchus acutus	Atlantic white-sided dolphin
15	Steno bredanensis	Rough-toothed dolphin, Steno
16	Delphinus delphis subsp.B	Baja neritic common dolphin, longbeaked common dolphin
17	Delphinus delphis subsp.A	Offshore common dolphin, shortbeaked common dolphin
18	Tursiops truncatus	Bottlenose dolphin, black porpoise, common porpoise
19	Cephalorhynchus heavisidii	Heaviside's dolphin
20	Cephalorhynchus hectori	Hector's dolphin, pied dolphin, white front dolphin
21	Grampus griseus	Risso's dolphin, gray grampus
22	Lagenorhynchus obliquidens	Pacific white-sided dolphin, lag, hookfin porpoise
23	Lagenorhynchus australis	Peale's dolphin, blackchin dolphin
24	Lagenorhynchus cruciger	Hourglass dolphin
25	Lagenorhynchus obscurus	Dusky dolphin
26	Lagenodelphis hosei	Fraser's dolphin, Sarawak dolphin
27	Lissodelphis borealis	Northern right whale dolphin
28	Lissodelphis peronii	Southern right-whale dolphin
29	Cephalorhynchus eutropia	Black dolphin, Chilean dolphin
30	Cephalorhynchus commersonii	Commerson's dolphin, piebald dolphin
31	Peponocephala electra	Melon-headed whale, Hawaiian/many-toothed blackfish, electra dolphin
32	Feresa attenuata	Pygmy killer whale, slender blackfish
33	Pseudorca crassidens	False killer whale
34	Globicephala sp.?	Unidentified pilot whale
35	Globicephala melas	Long-finned pilot whale, Atlantic pilot whale, blackfish, pothead
36	Globicephala macrorhynchus	Short-finned pilot whale, blackfish, pothead
37	Orcinus orca	Killer whale
38	Sousa chinensis	Indo-Pacific hump-backed dolphin, white dolphin
39	Sousa teuszii	Atlantic hump-backed dolphin
40	Phocoena phocoena	Harbor porpoise, herring hog
41	Phocoena sinus	Vaquita, Gulf of California harbor porpoise
42	Phocoena spinipinnis	Burmeister's porpoise, black porpoise
43	Neophocaena phocaenoides	Black finless porpoise
44	Phocoenoides dalli	Dall's porpoise
45	Delphinapterus leucas	White whale, beluga, belukha, sea canary
46	Physeter macrocephalus	Sperm whale
47	Kogia breviceps	Pygmy sperm whale
48	Kogia simus	Dwarf sperm whale
49	Ziphiid ?	Unidentified beaked whale
50	Hyperoodon planifrons	Southern bottlenose whale, flathead bottlenose whale

Appendix 6

C O D E T A B L E 4 b
Research Vessel / Life History
Cetacean Species Codes

- - numeric - -

SPEC CODE	SCIENTIFIC NAME	STANDARD COMMON NAME, OTHER COMMON NAMES
51	Mesoplodon sp.?	Unidentified Mesoplodon
52	Mesoplodon carlhubbsi	Hubb's beaked whale, archbeak whale
53	Mesoplodon hectori	Hector's beaked whale
54	Mesoplodon bowdoini	Andrew's beaked whale, deepcrest whale
55	Mesoplodon europaeus	Gervais' beaked whale, Antillean beaked whale
56	Mesoplodon bidens	Sowerby's beaked whale
57	Mesoplodon ginkgodens	Ginkgo-toothed beaked whale
58	Mesoplodon grayi	Gray's beaked whale
59	Mesoplodon densirostris	Blaineville's beaked whale, dense-beaked, tropical beaked whale
60	Mesoplodon layardii	Strap-toothed whale
61	Ziphius cavirostris	Cuvier's beaked whale, gooze-beaked whale
62	Berardius arnuxii	Arnoux's beaked whale, southern giant bottlenose whale
63	Berardius bairdii	Baird's beaked whale, southern giant bottlenose whale
64	Tasmacetus shepherdi	Shepherd's beaked whale
65	Mesoplodon pacificus	Longman's beaked whale, Indo-Pacific beaked whale
66	Eubalaena glacialis	Northern right whale
67	Balaena mysticetus	Bowhead whale
68	Caperea marginata	Pygmy right whale
69	Eschrichtius robustus	Gray whale
70	Balaenoptera sp.?	Unidentified Rorqual
71	Balaenoptera acutorostrata	Minke whale
72	Balaenoptera edeni	Bryde's whale
73	Balaenoptera borealis	Sei whale
74	Balaenoptera physalus	Fin whale
75	Balaenoptera musculus	Blue whale
76	Megaptera novaeangliae	Humpback whale
77		Unidentified dolphin or porpoise
78		Unidentified small whale
79		Unidentified large whale
80	Kogia simus/breviceps	Unidentified Kogia - dwarf or pygmy sperm whale
81	Mesoplodon steinegeri	Steinger's beaked whale, sabertooth, Bering Sea beaked bhale
82	Mesoplodon mirus	True's Beaked Whale
83	Mesoplodon sp.A	Unnamed beaked whale
84	Hyperoodon ampullatus	Northern Bottlenose, North Atlantic bottlenose whale
85	Monodon monoceros	Narwhal, sea unicorn
86	Eubalaena australis	Southern right whale
87	Pontoporia blainvillei	Franciscana, La Plata dolphin
88	S. longirostris centroamericana	Central American spinner dolphin, Costa Rican spinner dolphin
89	Stenella attenuata/plagidon	Unidentified spotted dolphin in Atlantic
90	Stenella attenuata subsp.?	Unidentified pantropical spotted dolphin, spotter porpoise
91	Stenella frontalis	Atlantic spotted dolphin, spotter porpoise
92	Platanista gangetica	Ganges susu, Ganges dolphin
93	Plantanista minor	Indus susu, Indus dolphin
94	Inia geoffrensis	Boto, Amazon river dolphin
95	Lipotes vexillifer	Baiji, Chinese river dolphin, whitefin dolphin
96		Unidentified cetacean
97		Unidentified object, possible marine mammal
98		Unidentified whale
99	Balaenoptera borealis/edeni	Rorqual identified as a Sei or Bryde's whale

Appendix 6

C O D E T A B L E 4 b

Pinniped Species Codes

SPEC CODE	SCIENTIFIC NAME	STANDARD COMMON NAME, OTHER COMMON NAMES....
PU	Pinnipedia	Unidentified Pinniped
UO	Otariinae	Unidentified Sea Lion
EJ	Eumetopias jubatus	Stellar Sea Lion
ZC	Zalophus californianus	California Sea Lion
UA	Arctocephalinae	Unidentified Fur Seal
AT	Arctocephalus townsendi	Guadalupe Fur Seal
CU	Callorhinus ursinus	Northern Fur Seal
US	Phociade	Unidentified Seal
MA	Mirounga angustirostris	Northern Elephant Seal
PV	Phoca vitulina	Harbor Seal

RECENT TECHNICAL MEMORANDUMS

Copies of this and other NOAA Technical Memorandums are available from the National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22167. Paper copies vary in price. Microfiche copies cost $4.50. Recent issues of NOAA Technical Memorandums from the NMFS Southwest Fisheries Science Center are listed below:

NOAA-TM-NMFS-SWFSC-185 Report of the two aerial surveys for marine mammals in California coastal waters utilizing a NOAA DeHavilland Twin Otter Aircraft March 9-April 7, 1991 and February 8-April 6, 1992.
J.V. CARRETTA and K.A. FORNEY
(September 1993)

186 The biology and population status of marine turtles in the North Pacific Ocean.
K.L. ECKERT
(September 1993)

187 Hawaiian monk seal observations at French Frigate Shoals, 1985.
J.J. ELIASON, J.R. HENDERSON, and M.A. WEBBER
(September 1993)

188 "Best" abundance estimates and best management: Why they are not the same.
B.L. TAYLOR
(October 1993)

189 Fishery interaction between the tuna longline and other pelagic fisheries of Hawaii.
R.A. SKILLMAN, C.H. BOGGS, and S.G. POOLEY
(October 1993)

190 Statistical guidelines for a pilot observer program to estimate turtle takes in the Hawaii longline fishery.
G.T. DiNARDO
(November 1993)

191 The Hawaiian monk seal on Laysan Island.
D.J. ALCORN and R.L. WESTLAKE
(December 1993)

192 Techniques for the preparation and examination of reproductive samples collected from dolphins in the eastern tropical Pacific.
P.A. AKIN, K.M. PELTIER, and R.B. MILLER
(December 1993)

193 A comparison of the recreational and commercial fisheries for lingcod (*Ophiodon elongatus*) off the Pacific coast of the United States, Along with a description of the recreational lingcod fishery.
K.R. SILBERBERG and P.B. ADAMS
(December 1993)

194 Economic effects of the United Nations moratorium on high seas driftnet fishing.
D.D. HUPPERT and T.W. MITTLEMAN
(December 1994)