June 2007

Developing a Geographic Information System Index for Historical Aerial Photographs

Benjamin Spaulding
University of Connecticut, benjamin.spaulding@uconn.edu

Sarah Stanwicks
University of Connecticut, sarah.stanwicks@uconn.edu

UCCGIA
Papers and Proceedings

Number 3

2007

Developing a Geographic Information System Index for Historical Aerial Photographs

Benjamin Spaulding
Department of Geography
University of Connecticut

Sarah Stanwicks
Homer Babbidge Library
University of Connecticut

Abstract: This paper describes the creation of a GIS database index to the collection of historical aerial photographs of Connecticut housed in the Map and Geographic Information Center in the Homer Babbidge Library at the University of Connecticut. The index allows patrons to search for scanned aerial photograph images for a specific location across multiple years and to retrieve digital scans from the Library server. Procedures for scanning and georeferencing the images, preparing metadata for the images, and creating the GIS database index are described.

Keywords: aerial photography, digital imagery, georeferencing, metadata, Connecticut

University of Connecticut
Center for Geographic Information and Analysis
Storrs, Connecticut

Advancing the use of geographic data and spatial analytic techniques at the University of Connecticut and in the region it serves

Cataloging Data
Spaulding, Benjamin
Developing a geographic information system index for historical aerial photographs / Benjamin Spaulding, Sarah Stanwicks
i, 8, 1, 14 p. : ill. ; 28 cm.
Includes bibliographical references.
(UCCGIA papers and proceedings ; no. 3)
1. Aerial photographs. 2. Geographic information systems. 3. Information storage and retrieval systems—Geography. 4. Metadata. 5. Connecticut—Aerial photographs. I. Stanwicks, Sarah. II. Homer Babbidge Library (University of Connecticut). Map and Geographic Information Center. III. University of Connecticut. Center for Geographic Information and Analysis.
G70.2 .U3 no. 3

Suggested Citation:
Spaulding, Benjamin and Stanwicks, Sarah. 2007. Developing a geographic information system index for historical aerial photographs. UCCGIA Papers and Proceedings, No. 3. Storrs, Connecticut: University of Connecticut Center for Geographic Information and Analysis.

Copies of this publication are available at the Map and Geographic Information Center, Homer Babbidge Library, University of Connecticut, Storrs, Connecticut, and through *Digital* Commons@UConn at http://digitalcommons.uconn.edu/uccgia_papers/3/

Table of Contents

Introduction

Historical aerial photographs are an important data source for engineers and environmental consultants, geographers and anthropologists, public officials, and private citizens who want to know what was at a certain location at a given time. One of the main uses of historical aerial photographs is measuring landscape change over time (Civco et al. 1986; Turner and Ruscher 1988; Rango and Havstad 2003; Laliberte et al. 2004). Over the past 75 years aerial photography, produced using a variety of methods, has helped define regions by providing accurate portrayals of the landscape. Early analysis of aerial photographs used stereoplotters and other tools to draw accurate maps (Falkner 1995). With the advent of geographic information systems (GIS) and the Internet, people can access spatial data in the form of aerial photographs from their homes and offices or through mobile devices.

The availability of aerial photography in Connecticut dates back to 1934 when the well-known Fairchild Aerial Survey was made for the entire state. Photographs from this survey have been used in research for many decades (Miller and Egler 1950; Rae 2001), and they are currently available to the public over the Internet through state and university Web sites (Connecticut State Library 2007; MAGIC 2007). Collections of historical aerial photography have grown in regional planning organization offices, university libraries, and state agencies where they have often been stored in file cabinets or in library stacks in microfiche or film negative formats. As with spatial data in most original formats, the integrity and condition of these early aerial photographs deteriorate over time due to handling (Connecticut State Library 2007).

Today aerial photographs are taken and stored digitally without the use of paper and film negatives. To preserve the large collections of historical aerial photographs, several steps are taken to protect the paper documents and to improve significantly the methods for searching for particular locations through time. Methods and practices for storing and distributing historical aerial photographs to the public vary greatly across the U.S., as the table in Appendix A shows. The information on state practices in Appendix A was compiled from a general Internet search of Web sites of state and university libraries and state-sponsored geospatial data centers. Some states, like Rhode Island and Delaware, support full-fledged Internet Map Servers (IMS) with complete georectified state aerial photography coverage. Other states, like Illinois and Virginia, have IMS that show center points of historical aerial photographs with each center point linking to the image which the user can then download.

However, few states have fully-developed location-based systems for finding historical aerial photographs. Typically, Web-based searches of aerial photograph collections require users to search first for a county or a town and then a neighborhood in the town to locate the aerial photographs of interest. These systems work best if the user is familiar with the place names in the area they are searching, making it hard for someone unfamiliar with these names to find specific locations. Alternatively, for many historical aerial photography collections, users of the images would have to be familiar with the large paper indexes used to organize the photos. At times, the search and retrieval process will be slow or ineffective for the user who is under time or budget constraints. A GIS makes it possible to query specific locations by pointing to them and retrieving information about what is located at the location. GIS technology provides the opportunity to use spatial analytic techniques to find all of the photographs in a collection that intersect with a user-specified location and to store

and retrieve the photographs and data describing them more effectively.

The purpose of this study was to develop a GIS database of the locations of the sites of historical aerial photographs in the collection of the Map and Geographic Information Center (MAGIC) of Homer Babbidge Library at the University of Connecticut. This paper describes the design and use of a database containing a record for each of the aerial photographs in the collection with attribute data describing the image. Using GIS techniques, the user can select a location and return a data table of corresponding photographs with links to the scanned images and complete Federal Geographic Data Committee (FGDC) metadata. The GIS techniques enable users to search for a set of photographs using a variety of descriptive attributes.

Historical Aerial Photography in the MAGIC Collection

MAGIC has approximately 35,000 paper aerial photographs of places in the state of Connecticut. These range from standard nine inch by nine inch paper sheets with scales from 1:1,000 to 1:20,000, to larger format photographs, which are 24 inches by 24 inches with a scale of 1:8,000. MAGIC's paper collection covers the period from the 1940s to 1995. The most recent photographs from 2004 are digital and are not available through MAGIC's Web site.

The aerial photographs stored at MAGIC are organized into three categories:

- Photos of towns;
- Photos of counties;
- Photos of east-west or north-south transects covering the entire state.

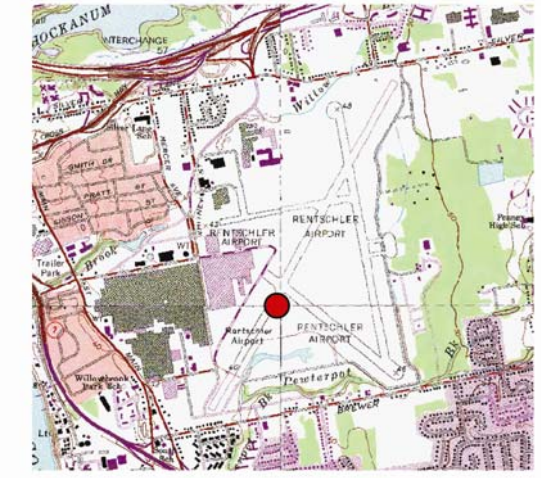
Towns are the basic units of local government in Connecticut and other southern New England states. They partition the state completely. Photographs collected from towns across various years are grouped by the U.S. Geological Survey (USGS) quadrangle in which the center point of the photograph is located. Most of the photographs organized in this fashion are from the mid-1940s, 1960s, and 1970s, and there are not complete sets of images for all towns for a given time. The second group includes photographs taken on a countywide basis, mainly under the auspices of the U.S. Department of Agriculture's Agricultural Stabilization Conservation Service (ASCS) during the 1950s and 1960s. The photographs in the rest of the collection, covering 1970, 1985, 1990, and 1995, are organized on a statewide basis based on flight lines moving across the entire state in an east-west direction (1970) or a north-south direction (1985, 1990, 1995). These flights were sponsored by various state agencies and photographs were taken as whole statewide sets.

Methodology

Creating Center Point Layers

A series of steps was taken to create an index of aerial photographs. The first step was to find the approximate center point of each of the photographs using GIS. To locate the center point of each of the photographs, a georeferenced image was used to match locations on the aerial photographs. A georeferenced USGS topographic map of the entire state of Connecticut, put together from the 7.5 minute quadrangle map series, had been created as part of the data development activities of MAGIC (MAGIC 2007). Using this image, the locations on an aerial photograph were matched to the georeferenced image and a center point was added at the photograph's approximate fiducial center.

Locating a center point for a photograph involved looking for distinguishing features on the photograph that could also be easily found on the map. The point corresponding to the distinguishing feature nearest to the center of the photograph was then identified as the "center" point on the corresponding GIS layer. In Figure 1, an airport is clearly visible in the center of the photograph. The georeferenced map also has the airport drawn and clearly visible. A simple point is added where the approximate center of the photograph is on the existing georeferenced map (Figure 1).



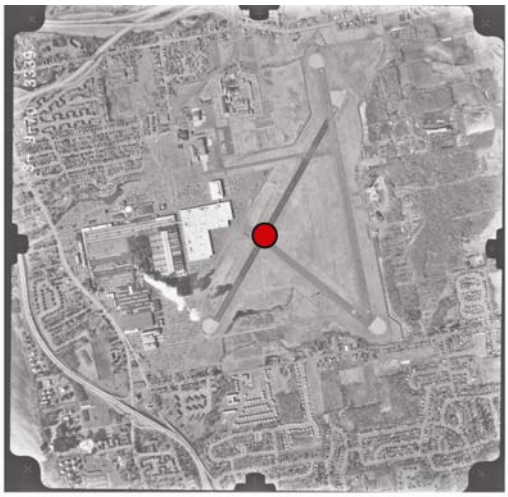


Figure 1. Matching a centerpoint of a photograph to a georeferenced map image.

When items in the series of images were geolocated, a pattern of center points arose due to the repeated coverage of 40 percent overlap on each photograph. Figure 2 shows an example of a collection of aerial photograph center points for photographs covering Hartford, Connecticut (Figure 2).

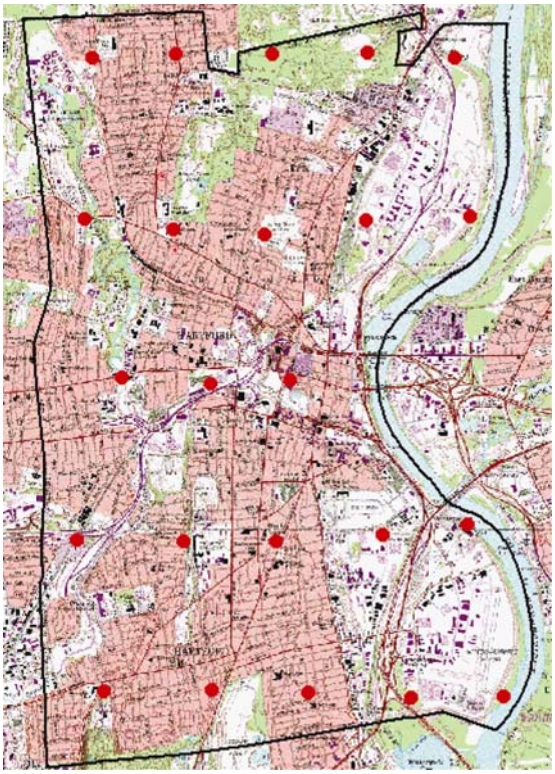


Figure 2. Center points for a group of aerial photographs covering Hartford, Connecticut.

A point geographic database in shapefile format was created in ArcView 3.3 for each photograph. Each record or row in the database table corresponds to a feature in the layer, and each column represents an attribute. Table 1 describes the attribute data collected for each aerial photograph center point (Table 1). The design was standardized for each of the aerial photograph shapefiles to ensure database integrity, facilitate metadata creation, and provide support for data distribution through future implementation of an Internet Map Server (IMS).

This approach also facilitated data entry. Because the attribute data for each of the photographs are similar, each photograph requires entry of data such as county, flight line, author, photograph date, and image identification number. Image identification numbers increment in a linear sequence. A database macro program was used to aid data input. The data developer set the rules for the database macro and used a command to fill the attributes of the individual records of georeferenced aerial photographs. The database macro created data for the ID, Homer_ID, Date, Projectcod, Countycode, Image_id, Scale, Author, and Hotlink fields. HOMER is the name of the University of Connecticut Libraries' online catalog. To establish a unique identification for each record and photograph, as well as aid in the Libraries' cataloging process, a barcode was added to both the photograph and to the corresponding attribute field in the database table.

Table 1. Center Point Shapefile Attributes

Attribute	Description
ID	Primary key of table
Homer_ID	Library identification number that links to library catalog
Date	Date photograph was taken
Projectcod	County_Year code describing the set the aerial photograph is part of
County	County where center point of photograph is located
Countycode	Library of Congress code for Connecticut County where center point of photograph is located
Image_id	Unique combination of flight line and image number
Scale	Scale of paper photograph
Author	Agency that sponsored photography
Hotlink	URL link to MrSID image on the server
Barcode	Barcode for library identification purposes
Town	Town where center point of photograph is located
Quad	USGS 7.5 minute quadrangle area where center point of photograph is located
Xul_cord	Westernmost longitude of the aerial photograph
Xlr_cord	Easternmost longitude of aerial photograph
Yul_cord	Northernmost latitude of the aerial photograph
Ylr_cord	Southernmost latitude of the aerial photograph

Not all of the attribute data were added through the database macro. Data for town, county, quad, and bounding box coordinates of the photographs were generated for each record using GIS techniques. To find the town, county, or USGS quadrangle in which a photograph's center point is located, a spatial join or overlay operation was used to join the aerial photograph center point index and Connecticut town, county, and quadrangle area shapefiles. Once the join operation was performed, the aerial photograph center points were associated with towns, counties, and quadrangle areas. The data attributes of these areas could then be joined to the set of attributes describing the aerial photograph center points.

There were a few aerial photograph center points that fell outside the borders of Connecticut and were not, therefore, assigned town or county identifiers as a result of the spatial join operation. The town and county locations of these center points were assigned manually based on distance to the nearest town or county. All of the aerial photograph center points fell within the boundaries of the USGS quadrangle areas covering Connecticut.

Creating Aerial Photograph Extent Shapefiles

Once the center points of the aerial photographs were identified, buffering was used to build an approximate representation of the aerial photography without rectifying

all the images. Coordinates were calculated for the bounding box of the image that resulted from the buffering process. A dimensionless point does not represent the entire coverage area of the photograph and its coordinates do not represent the extent of the image. With the scale of the aerial photographs known and with the data projected in Connecticut State Plane North American Datum (NAD) 1983 feet, the approximate coverage area could be derived by creating a square buffer representing the square footage of the aerial photograph.

The square buffer tool, added to the GIS application as an extension downloaded from the ESRI Web site (Scheitlin 2000), was used to create a new polygon shapefile layer for each of the aerial photograph center point shapefiles. With the projection defined and the distance units set to feet (to conform with the map units of the Connecticut State Plane NAD 1983 projection), an aerial photograph was displayed in the GIS. A square was drawn to cover the extent of the aerial photograph. The measurement dimensions of the graphic were used to set the parameters for creating the square buffer. The extension created the square buffers in the polygon shapefile in the same order as the center points are found in the point shapefile attribute table. In the resulting polygon shapefile, a unique square represented each of the center points. The new polygon layer showed the approximate coverage area of each image and the areas of overlap among images.

With the extent of each photograph modeled by creating the square buffer, the *x*- and *y*-coordinates of each photograph's approximate bounding rectangle were calculated. Coordinates were first calculated based on the Connecticut State Plane NAD 1983 projection, and then calculated in decimal degrees. The bounding box coordinate data values were then joined to the attribute table of the original aerial photograph center point shapefile using the common ID field as the primary key.

Once all of the square buffer shapefiles approximating the extent of each aerial photograph were created, all of the aerial photography polygon layers were merged into a single layer of square buffers representing all of the aerial photographs in the collection. Because the database table design was the same for each shapefile, it was easy to create a single file to serve as a master index of aerial photographs.

This master index data layer of historical aerial photography can be queried in GIS to find all aerial photographs for a particular year, town, county, or author. The query can be based on attributes or be performed as a spatial query in GIS. To perform the query as an attribute query, the user would enter a name or year in the query string. To perform the query as a spatial query, the user would draw a rectangle around the area of interest and the query would return all of the select aerial photograph square buffer shapes that interested the rectangle.

Scanning Aerial Photographs

While the index database of aerial photographs was being created, the paper photographic images were scanned and converted into compressed mulitresolution seamless image database (MrSID) image files. MrSID is a wavelet compression program that allows for large files to be easily viewed without loss in resolution or clarity.

To link the scanned image to the record in the GIS database, a column was added to the attribute table. This column was populated with a link to the corresponding image located on a server. Scanning the images served a dual purpose: 1) it made the images easier to access for users because they could be accessed over the

Internet; and 2) it preserved the paper aerial photographs for future use.

All the photographs were scanned using a Microtek ScanMaker 9800XL scanner at 300 dots per inch (dpi) at 100 percent size in Tagged Image File Format (TIFF). At this resolution, the details of the image can be seen at a high zoom level but file sizes are manageable for the storage system. A dpi of 300 was chosen because it provides a good reference image for a majority of MAGIC's patrons. Advanced users who need higher quality scans can obtain them from MAGIC upon request.

MrSID Compress v1.2 was used to compress the scanned images. The target compression was 20 with 4 zoom levels and a target thumbnail of 5000 pixels.

As noted, the original TIFF scans were saved on compact discs (CDs) for preservation purposes. This was accomplished using Roxio Easy CD Creator 5. On average, 20 images could be stored on a single CD. A database describing each CD was also created. The CDs were archived to the database using Cdarchive2.exe, a program developed at MAGIC. The program creates a record in the database for each CD containing the CD's name and directory of files. These two components were used in the process of creating metadata for the project.

When patrons come to MAGIC, the images saved on the server are the ones that are viewed. The images stored on CDs are only for preservation purposes. In the future, when users access the images remotely via an IMS, images will appear in MrSID format.

Creating Metadata

Metadata, data about data, are crucial to researchers, allowing them to verify the validity and accuracy of the data they will be using in their research. Metadata standards set by the Federal Geographic Data Committee (FGDC 1998) were used in this project for each of the individual aerial photographs in the collection. The attributes in the center point shapefiles' attribute tables were designed, in part, to correspond to the content requirements of FDGC-compliant metadata.

The metadata entry process was automated using computer code developed through a project initiated by MAGIC in conjunction with the Library's Information Technology Services (ITS) department. This code was used to create xml, htm, and sgml files for all of the aerial photographs in the collection. The Perl script pulled data from a set of relational databases. A sample of the script is provided in Appendix B (Appendix B). The data tables used to create the metadata include the information from the center point shapefile attribute table data, the directory information for each of the CDs created, and catalog information about the aerial photographs.

Results and Conclusion

With the advent of GIS, MAGIC has been able to streamline the process of locating photographs across multiple years for a specific location using a database of geo-located aerial photographs. Once the photographs are found using the GIS database, digital scans are extracted from the server for the patron, turning the process of locating photographs for a specific location from what used to be an hour-long search and retrieval process to a minutes-long process. Users can now come to MAGIC and have staff find all of the photographs in the collection that cover a particular location, see the digital images, and look at the metadata.

Users can also view the index maps through a Web site created for this purpose (MAGIC – Connecticut Historical Aerial Photography, 2007). County maps showing

aerial photograph center points for individual years from 1951 to 1995 can be downloaded in .pdf format. The center point databases can also be downloaded in shapefile or geodatabase format.

Future development of this system might include an online user interface where patrons can search by address, town, quadrangle, coordinates, or county to find the set of corresponding photographs. This system would then direct the user to links to the online photography which would be available for download. Also, these images could be georectified, creating the world files needed so that images can be brought directly into a GIS to be analyzed. As the system is currently set up, none of the images are georectified. MAGIC did not have the time or resources to perform this task.

This system could also be created as a GIS extension with the data packaged into an executable program that could be distributed to library patrons. The historical aerial photography indexes would be of great use to environmental consulting firms doing historical site assessment, planning officials, or historians working on research projects.

References

Connecticut State Library. 2007. Research Guide to Aerial Photographs at the Connecticut State Library. Online. http://www.cslib.org/aerials [March 20, 2007].

Civco, D. L., W. C. Kennard, and M. W. Lefor. 1986. Changes in Connecticut salt-marsh vegetation as revealed by historical aerial photographs and computer-assisted cartographics. *Environmental Management* 10(2): 229-239.

Falkner, E. 1995. *Aerial Mapping: Methods and Applications*. Ann Arbor: Lewis Publishers.

Federal Geographic Data Committee. (1998). FGDC-STD-001-1998. *Content Standard for Digital Geospatial Metadata* (revised June 1998). Washington, D.C.: Federal Geographic Data Committee.

Laliberte, A. S., A. Rango, K. M. Havstad, J.F. Paris, R.F. Beck, R. McNeely, and A. L. Gonzalez. 2004. Object-oriented image analysis for mapping shrub encroachment from 1937 to 2003 in southern New Mexico. *Remote Sensing of Environment* 93(1-2):198-210.

MAGIC (Map and Geographic Information Center). 2007. University of Connecticut Homer Babbidge Library. Online. http://magic.lib.uconn.edu [March 20, 2007].

MAGIC – Connecticut Historical Aerial Photography. 2007. University of Connecticut Homer Babbidge Library. Online. http://magic.lib.uconn.edu/adimg_project/CHAP/index.htm [November 7, 2007].

Miller, W. R., and F. E. Egler. 1950. Vegetation of the Wequetequock-Pawcatuck tidal-marshes, Connecticut. *Ecological Monographs* 20(2):143-172.

Rae D. W. 2001. Viacratic America: Plessy on foot v. Brown on wheels. *Annual Review of Political Science* 4(1): 417-438.

Rango, A., and K. Havstad. 2003. The utility of historical aerial photographs for detecting and judging the effectiveness of rangeland remediation treatments. *Environmental Practices* 5(2):107-118.

Scheitlin, R. J. 2000. Square buffer wizard. ESRI Support Center. Online. http://arcscripts.esri.com/details.asp?dbid=11530 [October 22, 2007].

Turner, M. G., and C. L. Ruscher. 1988. Changes in landscape patterns in Georgia, USA. *Landscape Ecology*, 1(4):241-251.

Appendix A
Historical Aerial Photograph Digital Collections by State

Table A-1. Survey of historical aerial photograph digital collections by state.

State	Aerial Photoraphy Internet Mapping	GIS Index of Aerial Photographs	DOQQs* Available	Place Name Index with Web Links	No Online Aerial Photography
Alabama				x	
Alaska					x
Arizona	x-current				
Arkansas					x
California	x				
Colorado	x	x			
Connecticut	x-current				
Delaware	x	x			
Florida	x	x			
Georgia	x-limited				
Hawaii			x		
Idaho		x			
Illinois	x	x		x	
Indiana	x			x	
Iowa	x				
Kansas				x	
Kentucky					x
Louisiana	x			x	
Maine	x			x	
Maryland	x				
Massachusetts	x		x		
Michigan				x	
Minnesota			x	x	
Mississippi			x		
Missouri			x		
Montana				x	
Nebraska			x		
Nevada				x	
New Hampshire	x				
New Jersey	x		x		
New Mexico					x
New York	x				
North Carolina				x	
North Dakota			x		
Ohio					x
Oklahoma			x		
Oregon				x	
Pennsylvania	x				
Rhode Island	x				
South Carolina			x		
South Dakota			x		
Tennessee			x		
Texas			x		
Utah			x		
Vermont					x
Virginia	x	x	x	x	
Washington			x		
West Virginia	x		x	x	
Wisconsin	x		x	x	

*DOQQ is an acronym for digital orthophoto quarter quad

Note: The information in this table was compiled from a general Internet search of Web sites of state and university libraries and state-sponsored geospatial data centers.

Appendix B
Metamagic.pl Perl Script for Metadata Creation

```
#!/usr/bin/perl -w

no warnings 'uninitialized';
use DBI;
use Fcntl qw(:DEFAULT :flock);
use LWP::Simple;
############################
### MAIN           ###
############################
&text_files;
&db_connect;
&parse_sid;
&read_sid;
&file_assign;
&error_check;
&file_write;
&file_process;
$end = `date`;print $end;
############################
### SUBROUTINES      ###
############################
############################
### text_files       ###
############################
# Create or clear the directory text_files.
sub text_files {
        if (-e "text_files/") {
                print "Checking text_files directory...\n";
              chdir("text_files/") or die "can't move to the directory text_files: $!";
              foreach $file (<*.*>) { # step through a list of all files
                 unlink($file) || warn "having trouble deleting $file: $!";
              }
        } else {
                print "Creating text_files directory...\n";
              mkdir("text_files",0600) || die "cannot mkdir text_files: $!";
        }
        chdir("/magic/working/perlcode") or die "can't chdir to /magic/working/perlcode: $!";
}
############################
### parse_sid        ###
############################
# This subroutine reads the file meta_magic.conf for the url to grab the data.
# It then parses the data and populates the filelist table that helps to build
# the meta table. Probably should write it to populate meta directly :|
sub parse_sid {
        my $url = "";
        &read_config;
        print "Getting data from $url\n";
        my $file = get("$url");
        chdir("/magic/working/perlcode/") or die "can't chdir to /magic/working/perlcode: $!";
        open (TMP, ">sid_file.txt") or die "can't open sid_file.txt: $!";
        print TMP $file or die "can't write sid_file.txt: $!";
        close (TMP) or die "can't close sid_file.txt: $!";
        open NOW, "sid_file.txt" or die "can't open sid_file.txt for reading: $!";
        $dbh->do("DELETE FROM filelist");
        print "Parsing sid_file.txt and populating filelist table...\n";
        while (<NOW>) {
                if (/^<img src="\/icons\/image2\.gif/ && /\.s\.sid<\/a/) { #had to make sure is a sid
and not metadata
                        my (undef, undef, undef, undef, $name, $date, $time, $size) = split /\s+/;
                        $name =~ s/href="([^"]+)".*/$1/;
```

```
                        my $out_name = $name;
                        # we don't want to use md extention anymore
                        # $out_name =~ s/s\.sid/md/;
                        my ($theme, $geo_arc, $image_id, $image_date, $scale, $author, $item,
$format, $extension) = split(/\./, $name);
                        $sth11=$dbh->prepare("SELECT Project_co FROM Project WHERE
Image_Date='$image_date' AND Item='$item'");
                        $sth11->execute();
                        while ( $array_ref = $sth11->fetchrow_arrayref ) {
                                $project_co = $array_ref->[0];
                        }
                        $sth12 = $dbh->prepare("INSERT INTO filelist VALUES ('$image_id',
'$project_co', '$image_date', '$date', '$size', '$out_name')");
                        $sth12->execute();
                }
        }
        ##########################
        ### read_config      ###
        ##########################
        sub read_config {
                open CFG, "meta_magic.conf" or die "can't open meta_magic.conf for reading: $!";
                while (<CFG>) {
                chomp;
                        if (/^http/) {  # found url entry...
                        $url = $_ ;
                        }
                }
                if ($url ne "") {
                # presume it's a good url!
                } else {
                print "No URL found in meta_magic.conf!\nEXITING!!!\n";
                exit;
                }
        }
}
##########################
### read_sid        ###
##########################
# This subroutine reads the directory /magic/working/perlcode/sid_output/sid on Urizen
# in order to populate the mysql db table sid with the appropriate information about
# each sid image. This depends on Patrick running Lizard Tech's program that pulls this
# information from the headers of the sid images and placing the files in the directory
# mentioned above.
sub read_sid {
      print "Reading sid directory and populating sid table...\n";
      chdir("/magic/working/perlcode/sid_output/sid") || die "cannot cd to
/magic/working/perlcode/sid_output/sid ($!)";
      $dbh->do("DELETE FROM Sid");
      while (defined($nextname = <*sid\.txt>)) {
            open (FILE, "$nextname");
            my ($file_name, $width, $height) = "";
            while (<FILE>) {
                  chomp;
                  if (/^Filename/) { $file_name = $_ }
                  if (/Image width/) { $width = $_ }
                  if (/Image height/) { $height = $_ }
            }
            close (FILE) or die "can't close file handle: $!";
            $file_name =~ s/Filename: //;
            my ($theme, $geo_arc, $image_id, $image_date, $scale, $author, $item, $format, $extension)
= split(/\./, $file_name);
            $width =~ s/\s+Image width: //;
            $height =~ s/\s+Image height: //;
            my $project_co = "";
            $sth110=$dbh->prepare("SELECT Project_co FROM Project WHERE
Image_Date='$image_date' AND Item='$item'");
```

```
            $sth110->execute();
            while ( $array_ref = $sth110->fetchrow_arrayref ) {
                  $project_co = $array_ref->[0];
            }
            $sth120 = $dbh->prepare("INSERT INTO Sid VALUES ('$image_id', '$project_co',
'$image_date', '$width', '$height')");
            $sth120->execute();
      }
}
##########################
### db_connect      ###
##########################
# Connect to the database!
sub db_connect {
         $begin = `date`;print $begin;
         $database = "fgdcmeta";
         $driver = "DBI:mysql";
         $user = "bkysela";
         $passwd = "foosball";
         $dbh = DBI->connect("$driver:database=$database", $user, $passwd) or die "can't connect to
fgdcmeta!: $!";
}
##########################
### file_assign     ###
##########################
# This subroutine creates the master database table meta from the
# tables in the fgdcmeta database. The table meta holds all the values
# used to create the text file that will be parsed by the mp program.
sub file_assign {
         $dbh->do("DELETE FROM meta");
         $sth = $dbh->prepare("INSERT INTO meta (Image_id, project_co, date, file_date, file_size,
file_name) SELECT * FROM filelist");
         $sth->execute();
         $sth->finish();
         $dbh->do("UPDATE meta,Chap SET meta.chap_date=Chap.date, meta.Hotlink=Chap.Hotlink,
meta.ul_x=Chap.ul_x, meta.ul_y=Chap.ul_y, meta.lr_x=Chap.lr_x, meta.lr_y=Chap.lr_y,
meta.Quadrangle=Chap.Quadrangle, meta.Homer_id=Chap.Homer_id, meta.barcode=Chap.barcode
WHERE meta.Image_id=Chap.Image_id AND meta.project_co=Chap.project_co");
         print "Got chap data!\n";
         $dbh->do("UPDATE meta,Sid SET meta.height=Sid.height, meta.width=Sid.width WHERE
meta.Image_id=Sid.Image_id AND meta.Project_co=Sid.Project_co");
         print "Got sid data!\n";

    # Currently only supported CHAP images in the format adimg.37800.01-234.1234.s20.usgs.1.s.sid
    # this assumes that you can take the first 35 characters to compare them
    # to support other length file names such as CCHAP adimg.37831.01-DPD10H-
114.1951.s20.pma.1.s.sid
    # in which case 35 characters is too few to get the number before the s
    # in the new method the file names will be broken down into section and compared section by section
    # this may add considerably more time, as it has to compare more objects, but should be well worth
    # the added accuracy...
    # Possible problem... to make up for the lack of a format (ie. .s.extention) on some county photos this
is
    # not being compared here. It may be important to compare it in other projects where it differs
    # Also some country photos have problems with dashes being inconsistant between the CD versions
and the
    # compressed versions. Taking these out should not make a difference but it is something to keep in
mind
    # if meta_magic is going to be used for some projects in the future that might differ
    # Update 8/17/04 James Ford

         # Old method
         # $sth = $dbh->prepare("UPDATE meta,Files SET meta.CDID=Files.CDID,meta.Size=Files.Size
WHERE left(Files.Name, 35) = left(meta.file_name, 35)");
         # New method
      $sth = $dbh->prepare("SELECT file_name FROM meta");
```

```
        $sth->execute();
      $sth->bind_col( 1, \$meta_file_name );
      while ($sth->fetchrow_array()) {
          ($m_theme, $m_geo_arc, $m_image_id, $m_image_date, $m_scale, $m_author, $m_item,
$m_format, $m_extension) = split(/\./, $meta_file_name);
          $m_image_id =~ s/-//g;
          $sth1 = $dbh->prepare("SELECT Name, CDID, Size FROM Files");
          $sth1->execute();
          $sth1->bind_col( 1, \$files_file_name );
          $sth1->bind_col( 2, \$files_CDID );
          $sth1->bind_col( 3, \$files_Size );
          while ($sth1->fetchrow_array()) {
              ($f_theme, $f_geo_arc, $f_image_id, $f_image_date, $f_scale, $f_author, $f_item,
$f_format, $f_extension) = split(/\./, $files_file_name);
              $f_image_id =~ s/-//g;
              if ($m_theme eq $f_theme && $m_geo_arc eq $f_geo_arc && $m_image_id eq $f_image_id
&& $m_image_date eq $f_image_date && $m_scale eq $f_scale && $m_author eq $f_author && $m_item
eq $f_item)
              {
                $sql = qq{ UPDATE meta SET CDID='$files_CDID', Size='$files_Size' WHERE
file_name='$meta_file_name' };
                $sth2 = $dbh->prepare( $sql );
                $sth2->execute();
                $sth2->finish();
                break;
              }
          }
          $sth1->finish();
        }
        $sth->finish();
        print "Got CDID and Size!\n";
        $sth45 = $dbh->prepare("UPDATE meta,CDs SET meta.CDName=CDs.CDName WHERE
meta.CDID=CDs.CDID");
        $sth45->execute();
        $sth45->finish();
        print "Got CD Name!\n";
        $dbh->do("UPDATE meta,Homer SET meta.Title=Homer.Title,
meta.CorporateName=Homer.CorporateName, meta.GeneralNote1=Homer.GeneralNote1,
meta.GeneralNote2=Homer.GeneralNote2, meta.GeneralNote3=Homer.GeneralNote3,
meta.RatioLinearHorizontal=Homer.RatioLinearHorizontal WHERE meta.Homer_id=Homer.Homer_id");
        print "Got Homer data!\nGetting town names...\n";
        $sth = $dbh->prepare("SELECT Project_co, Image_id from meta");
        $sth->execute();
        $sth->bind_col( 1, \$project_co );
        $sth->bind_col( 2, \$image_id );
        while ($sth->fetchrow_array()) {
                $sth1 = $dbh->prepare("SELECT DISTINCT name FROM Chaptown WHERE Project_co =
'" . $project_co . "' AND Image_id = '" . $image_id ."'");
                $sth1->execute();
                $string = "";
                while (@array = $sth1->fetchrow_array()) {
                        for ($i=0;$i<=$#array;$i++) {
                                $string .= "$array[$i], ";
                        }
                }
                $sth1->finish();
                $sth2 = $dbh->prepare("UPDATE meta SET Name='" . $string . "' WHERE Image_id = '"
. $image_id . "' AND Project_co = '" . $project_co . "'");
                $sth2->execute();
                $sth2->finish();
        }
        $sth->finish();
        print "Got town names!\n";
}
##########################
### error_check      ###
```

```
##########################
# This subroutine checks for missing values in meta.
sub error_check {
        #Clear the error tables
        $dbh->do("DELETE FROM cd_errors");
        $dbh->do("DELETE FROM chap_errors");
        $dbh->do("DELETE FROM homer_errors");
        $dbh->do("DELETE FROM sid_errors");
        $dbh->do("DELETE FROM other_errors");

        $sth6 = $dbh->prepare("SELECT CorporateName, chap_date, Title, project_co,
Image_id,Hotlink, GeneralNote1, GeneralNote2, GeneralNote3, Homer_id, Barcode, ul_x, ul_y, lr_x, lr_y,
Quadrangle, Name, file_size, RatioLinearHorizontal, file_date, width, height, Size, CDName, CDID,
file_name FROM meta");
        $sth6->execute();
        $sth6->bind_col( 1, \$errhomer_CorporateName);
        $sth6->bind_col( 2, \$errchap_date);
        $sth6->bind_col( 3, \$errhomer_Title);
        $sth6->bind_col( 4, \$errproject_co );
        $sth6->bind_col( 5, \$errimage_id);
        $sth6->bind_col( 6, \$errchap_hotlink);
        $sth6->bind_col( 7, \$errhomer_GeneralNote1);
        $sth6->bind_col( 8, \$errhomer_GeneralNote2);
        $sth6->bind_col( 9, \$errhomer_GeneralNote3);
        $sth6->bind_col( 10, \$errhomer_id);
        $sth6->bind_col( 11, \$errbarcode);
        $sth6->bind_col( 12, \$errchap_Ul_x);
        $sth6->bind_col( 13, \$errchap_Ul_y);
        $sth6->bind_col( 14, \$errchap_Lr_x);
        $sth6->bind_col( 15, \$errchap_Lr_y);
        $sth6->bind_col( 16, \$errchap_Quadrangle);
        $sth6->bind_col( 17, \$errName);
        $sth6->bind_col( 18, \$errfile_size);
        $sth6->bind_col( 19, \$errRatioLinearHorizontal);
        $sth6->bind_col( 20, \$errfile_date);
        $sth6->bind_col( 21, \$errsid_width);
        $sth6->bind_col( 22, \$errsid_height);
        $sth6->bind_col( 23, \$errSize);
        $sth6->bind_col( 24, \$errCDName);
        $sth6->bind_col( 25, \$errCDID);
        $sth6->bind_col( 26, \$errfile_name);
        chdir("/magic/working/perlcode/");
        sysopen (ERR, "error_log.txt", O_RDWR | O_CREAT) or die "can't open error_log.txt: $!";
        print "Printing error_log.txt...\n";
        while ($sth6->fetchrow_array()) {

                #empty error arrays
                @cd = undef;
                @chap = undef;
                @homer = undef;
                @sid = undef;
                @other = undef;

                print ERR "$errfile_name\n";
                print ERR "***************Problems***************\n";
                if (!defined($errSize)) {
                        print ERR "No value for Size\n" or die "can't write file: $!";
                        push( @cd, "1" );
                }
                if (!defined($errCDName)) {
                        print ERR "No value for CDName\n" or die "can't write file: $!";
                        push( @cd, "2" );
                }
                if (!defined($errCDID)) {
                        print ERR "No value for CDID\n" or die "can't write file: $!";
                        push( @cd, "3" );
```

```
}
if (!defined($errchap_date)) {
        print ERR "No value for chap_date\n" or die "can't write file: $!";
        push( @chap, "1" );
}
if (!defined($errchap_hotlink)) {
        print ERR "No value for chap_hotlink\n" or die "can't write file: $!";
        push( @chap, "2" );
}
if (!defined($errbarcode)) {
        print ERR "No value for barcode\n" or die "can't write file: $!";
        push( @chap, "3" );
}
if ($errbarcode == 0) {
        print ERR "Barcode set to zero\n" or die "can't write file: $!";
        push( @chap, "4" );
}
if (!defined($errchap_Ul_x)) {
        print ERR "No value for chap_Ul_x\n" or die "can't write file: $!";
        push( @chap, "5" );
}
if (!defined($errchap_Ul_y)) {
        print ERR "No value for chap_Ul_y\n" or die "can't write file: $!";
        push( @chap, "6" );
}
if (!defined($errchap_Lr_x)) {
        print ERR "No value for chap_Lr_x\n" or die "can't write file: $!";
        push( @chap, "7" );
}
if (!defined($errchap_Lr_y)) {
        print ERR "No value for chap_Lr_y\n" or die "can't write file: $!";
        push( @chap, "8" );
}
if (!defined($errchap_Quadrangle)) {
        print ERR "No value for chap_Quadrangle\n" or die "can't write file: $!";
        push( @chap, "9" );
}
if (!defined($errhomer_CorporateName)) {
        print ERR "No value for homer_CorporateName\n" or die "can't write file: $!";
        push( @homer, "1" );
}
if (!defined($errhomer_Title)) {
        print ERR "No value for homer_Title\n" or die "can't write file: $!";
        push( @homer, "2" );
}
if (!defined($errhomer_GeneralNote1)) {
        print ERR "No value for homer_GeneralNote1\n" or die "can't write file: $!";
        push( @homer, "3" );
}
if (!defined($errhomer_GeneralNote2)) {
        print ERR "No value for homer_GeneralNote2\n" or die "can't write file: $!";
        push( @homer, "4" );
}
if (!defined($errhomer_GeneralNote3)) {
        print ERR "No value for homer_GeneralNote3\n" or die "can't write file: $!";
        #push( @homer, "5" );
}
if (!defined($errhomer_id)) {
        print ERR "No value for homer_id\n" or die "can't write file: $!";
        push( @homer, "6" );
}
if (!defined($errfile_size)) {
        print ERR "No value for file_size\n" or die "can't write file: $!";
        push( @sid, "1" );
}
if (!defined($errRatioLinearHorizontal)) {
```

```
                print ERR "No value for RatioLinearHorizontal\n" or die "can't write file: $!";
                push( @sid, "2" );
        }
        if (!defined($errfile_date)) {
                print ERR "No value for file_date\n" or die "can't write file: $!";
                push( @sid, "3" );
        }
        if (!defined($errsid_width)) {
                print ERR "No value for sid_width\n" or die "can't write file: $!";
                push( @sid, "4" );
        }
        if (!defined($errsid_height)) {
                print ERR "No value for sid_height\n" or die "can't write file: $!";
                push( @sid, "5" );
        }
        if (!defined($errName)) {
                print ERR "No value for town name\n" or die "can't write file: $!";
                push( @other, "1" );
        }
        if (!defined($errproject_co)) {
                print ERR "No value for project_co\n" or die "can't write file: $!";
                push( @other, "2" );
        }
        if (!defined($errimage_id)) {
                print ERR "No value for image_id\n" or die "can't write file: $!";
                push( @other, "3" );
        }
        print ERR "\n\n";

        #put stuff in DB if there are errors
        if (!defined(@cd)) { $cdsize = 0; }
        else { $cdsize = scalar @cd; }
        if (!defined(@chap)) { $chapsize = 0; }
        else { $chapsize = scalar @chap; }
        if (!defined(@homer)) { $homersize = 0; }
        else { $homersize = scalar @homer; }
        if (!defined(@sid)) { $sidsize = 0; }
        else { $sidsize = scalar @sid; }
        if (!defined(@other)) { $othersize = 0; }
        else { $othersize = scalar @other; }

        $sth7 = $dbh->prepare("SELECT project_co FROM meta WHERE file_name = '" .
$errfile_name . "'");
        $sth7->execute();
        $sth7->bind_col( 1, \$proj);
        while ($sth7->fetchrow_array()) {

        if (!defined($proj)) { $proj = "Not Found"; }

        if ($cdsize > 0) {
        $sth = $dbh->prepare("INSERT INTO cd_errors VALUES ('$errfile_name', '$cdsize',
'$proj')");
    $sth->execute();
        $sth->finish();
        }
        if ($chapsize > 0) {
        $sth = $dbh->prepare("INSERT INTO chap_errors VALUES ('$errfile_name', '$chapsize',
'$proj')");
    $sth->execute();
        $sth->finish();
        }
        if ($homersize > 0) {
        $sth = $dbh->prepare("INSERT INTO homer_errors VALUES ('$errfile_name',
'$homersize', '$proj')");
    $sth->execute();
        $sth->finish();
```

```
                }
                if ($sidesize > 0) {
                $sth = $dbh->prepare("INSERT INTO sid_errors VALUES ('$errfile_name', '$sidsize',
'$proj')");
        $sth->execute();
                $sth->finish();
                }
                if ($othersize > 0) {
                $sth = $dbh->prepare("INSERT INTO other_errors VALUES ('$errfile_name',
'$othersize', '$proj')");
        $sth->execute();
                $sth->finish();
                }
                }
                $sth7->finish();
        }
        close (ERR) or die "can't close error_log.txt: $!";
        $sth6->finish;
}
##########################
### file_write      ###
##########################
# This subroutine writes the text file for each image by creating
# a text file for each record returned by the query to the meta table.
sub file_write {
        $sth3 = $dbh->prepare("SELECT CorporateName, chap_date, Title, Project_co, Image_id,
Hotlink, GeneralNote1, GeneralNote2, GeneralNote3, homer_id, barcode, ul_x, ul_y, lr_x, lr_y,
Quadrangle, Name, file_size, RatioLinearHorizontal, file_date, width, height, Size, CDName, CDID,
file_name FROM meta");
        $sth3->execute();
        $sth3->bind_col( 1, \$homer_CorporateName);
        $sth3->bind_col( 2, \$chap_date);
        $sth3->bind_col( 3, \$homer_Title);
        $sth3->bind_col( 4, \$project_co );
        $sth3->bind_col( 5, \$image_id);
        $sth3->bind_col( 6, \$chap_hotlink);
        $sth3->bind_col( 7, \$homer_GeneralNote1);
        $sth3->bind_col( 8, \$homer_GeneralNote2);
        $sth3->bind_col( 9, \$homer_GeneralNote3);
        $sth3->bind_col( 10, \$homer_id);
        $sth3->bind_col( 11, \$barcode);
        $sth3->bind_col( 12, \$chap_Ul_x);
        $sth3->bind_col( 13, \$chap_Ul_y);
        $sth3->bind_col( 14, \$chap_Lr_x);
        $sth3->bind_col( 15, \$chap_Lr_y);
        $sth3->bind_col( 16, \$chap_Quadrangle);
        $sth3->bind_col( 17, \$Name);
        $sth3->bind_col( 18, \$file_size);
        $sth3->bind_col( 19, \$RatioLinearHorizontal);
        $sth3->bind_col( 20, \$file_date);
        $sth3->bind_col( 21, \$sid_width);
        $sth3->bind_col( 22, \$sid_height);
        $sth3->bind_col( 23, \$Size);
        $sth3->bind_col( 24, \$CDName);
        $sth3->bind_col( 25, \$CDID);
        $sth3->bind_col( 26, \$file_name);
        print "Printing text files...\n";
        while ($sth3->fetchrow_array()) {
        if (defined($chap_date)) {
                $chap_date = &fix_chapdate($chap_date);
        } else {
                $chap_date = "";
        }
        if (defined($file_date)) {
                $file_date = &fix_filedate($file_date);
        } else {
```

```
                $file_date = "";
        }
        if (defined($Size)) {
                $kb = $Size / 1024; #Turn size on CD into KiloBytes from bytes
                $kb = sprintf("%0.2f", $kb); #only display two decimal points
        } else {
                $kb = "";
        }
        $chap_date_long = $chap_date;
        $now = `date +%F`;
        $file = "Identification_Information: \n";
        $file .= ("  Citation:\n");
        $file .= ("    Citation_Information:\n");
        $file .= ("      Originator: " . $homer_CorporateName . "\n");
        $file .= ("      Publication_Date: " . $chap_date . "\n");
        $file .= ("      Title: " . $homer_Title . " " . $chap_date_long . "; Project " . $project_co . "; Image
ID " . $image_id . "\n");
        $file .= ("      Geospatial_Data_Presentation_Form: remote-sensing image\n");
        $file .= ("      Online_Linkage: " . $chap_hotlink . "\n");
        $file .= ("  Description:\n");
        $file .= ("    Abstract: This aerial photograph is from the collections of historical aerial
photography\n");
        $file .= ("      at the Map and Geographic Information Center at the Homer Babbidge Library,
University\n");
        $file .= ("      of Connecticut. These photographs are being collected over time from a variety of
sources.\n");
        $file .= ("      They represent the changing landscape of Connecticut. The photographs have been
geo-located,\n");
        $file .= ("      but not geo-corrected or geo-rectified.\n");
        $file .= ("    Purpose: These photographs are provided for reference use.\n");
        $file .= ("    Supplemental_Information: " . $homer_GeneralNote1 . " " . $homer_GeneralNote2 .
" " . $homer_GeneralNote3 . " " . $project_co . " Homer#: " . $homer_id . " Barcode#: " . $barcode .
"\n");
        $file .= ("  Time_Period_of_Content:\n");
        $file .= ("    Time_Period_Information:\n");
        $file .= ("      Single_Date/Time: \n");
        $file .= ("        Calendar_Date: " . $chap_date . "\n");
        $file .= ("    Currentness_Reference: publication date\n");
        $file .= ("  Status: \n");
        $file .= ("    Progress: Complete\n");
        $file .= ("    Maintenance_and_Update_Frequency: None\n");
        $file .= ("  Spatial_Domain: \n");
        $file .= ("    Bounding_Coordinates: \n");
        $file .= ("      West_Bounding_Coordinate: " . $chap_Ul_x . "\n");
        $file .= ("      East_Bounding_Coordinate: " . $chap_Ul_y . "\n");
        $file .= ("      North_Bounding_Coordinate: " . $chap_Lr_x . "\n");
        $file .= ("      South_Bounding_Coordinate: " . $chap_Lr_y . "\n");
        $file .= ("  Keywords: \n");
        $file .= ("    Theme: \n");
        $file .= ("      Theme_Keyword_Thesaurus: LCSH\n");
        $file .= ("      Theme_Keyword: Remote-sensing images\n");
        $file .= ("      Theme_Keyword: Aerial photographs\n");
        $file .= ("    Place: \n");
        $file .= ("      Place_Keyword_Thesaurus: GNIS\n");
        $file .= ("      Place_Keyword: " . $chap_Quadrangle . " 7.5 min topographic quadrangle\n");
        if (defined($Name)) {
                my @towns = ();
                @towns = split(/,\s?/, $Name);
                for ($i=0;$i<=$#towns;$i++) {
                        $file .= ("      Place_Keyword: $towns[$i]\n");
                }
        }

        $file .= ("    Stratum: \n");
        $file .= ("      Stratum_Keyword_Thesaurus: None\n");
        $file .= ("      Stratum_Keyword: None\n");
```

```
$file .= ("    Temporal: \n");
$file .= ("      Temporal_Keyword_Thesaurus: None\n");
$file .= ("      Temporal_Keyword: None\n");
$file .= ("  Access_Constraints: None\n");
$file .= ("  Use_Constraints: None\n");
$file .= ("  Point_of_Contact:\n");
$file .= ("    Contact_Information:\n");
$file .= ("      Contact_Organization_Primary:\n");
$file .= ("        Contact_Organization:\n");
$file .= ("          Map and Geographic Information Center,\n");
$file .= ("          Homer Babbidge Library, Connecticut.\n");
$file .= ("          University of Connecticut.\n");
$file .= ("          Storrs, CT.\n");
$file .= ("      Contact_Address:\n");
$file .= ("        Address_Type: Mailing and Physical Address\n");
$file .= ("        Address: Map and Geographic Information Center U-5m\n");
$file .= ("        Address: Homer Babbidge Library, Connecticut.\n");
$file .= ("        Address: 369 Fairfield Road.\n");
$file .= ("        City: Storrs\n");
$file .= ("        State_or_Province: CT\n");
$file .= ("        Postal_Code: 06269-1005\n");
$file .= ("      Contact_Voice_Telephone: (860) 486-4589\n");
$file .= ("      Contact_Facsimile_Telephone: (860) 486-3593\n");
$file .= ("      Contact_Electronic_Mail_Address: patrick.mcglamery\@uconn.edu\n");
$file .= ("      Hours_of_Service: 9-12, 1-5 M-F, 18:30-21:30 M-W\n");
$file .= ("      Contact_Instructions: Unavailable\n");
$file .= ("  Native_Data_Set_Environment: sid file size= " . $file_size . "\n");
$file .= ("Data_Quality_Information:\n");
$file .= ("  Attribute_Accuracy:\n");
$file .= ("    Attribute_Accuracy_Report: None\n");
$file .= ("  Logical_Consistency_Report: None\n");
$file .= ("  Completeness_Report: None\n");
$file .= ("  Positional_Accuracy:\n");
$file .= ("    Horizontal_Positional_Accuracy:\n");
$file .= ("      Horizontal_Positional_Accuracy_Report: Not determined\n");
$file .= ("  Lineage:\n");
$file .= ("    Source_Information:\n");
$file .= ("      Source_Citation:\n");
$file .= ("        Citation_Information:\n");
$file .= ("          Originator: " . $homer_CorporateName . "\n");
$file .= ("          Title: " . $project_co . " " . $image_id . "\n");
$file .= ("          Publication_Date: " . $chap_date . "\n");
$file .= ("      Source_Scale_Denominator: " . $RatioLinearHorizontal . "\n");
$file .= ("      Type_of_Source_Media: Paper\n");
$file .= ("      Source_Time_Period_of_Content:\n");
$file .= ("        Time_Period_Information:\n");
$file .= ("          Single_Date/Time:\n");
$file .= ("            Calendar_Date: " . $chap_date . "\n");
$file .= ("        Source_Currentness_Reference: None\n");
$file .= ("      Source_Citation_Abbreviation: None\n");
$file .= ("      Source_Contribution: None\n");
$file .= ("    Process_Step:\n");
$file .= ("      Process_Description: The aerial photograph was scanned on a HP ScanJet 4c/T\n");
$file .= ("        at a resolution of 300 dpi. No contrast balancing was performed. The image\n");
$file .= ("        was saved as a TIFF file.\n");
$file .= ("      Source_Used_Citation_Abbreviation: None\n");
$file .= ("      Process_Date: " . $file_date . "\n");
$file .= ("      Process_Contact:\n");
$file .= ("        Contact_Information:\n");
$file .= ("          Contact_Organization_Primary:\n");
$file .= ("            Contact_Organization:\n");
$file .= ("              Map and Geographic Information Center,\n");
$file .= ("              Homer Babbidge Library, Connecticut.\n");
$file .= ("              University of Connecticut.\n");
$file .= ("              Storrs, CT.\n");
$file .= ("          Contact_Address:\n");
```

```
        $file .= ("          Address_Type: Mailing and Physical Address\n");
        $file .= ("          Address: Map and Geographic Information Center U-5m\n");
        $file .= ("          Address: Homer Babbidge Library, Connecticut.\n");
        $file .= ("          Address: 369 Fairfield Road.\n");
        $file .= ("          City: Storrs\n");
        $file .= ("          State_or_Province: CT\n");
        $file .= ("          Postal_Code: 06269-1005\n");
        $file .= ("        Contact_Voice_Telephone: (860) 486-4589\n");
        $file .= ("   Process_Step:\n");
        $file .= ("     Process_Description: The TIFF file was compressed using MrSID Compression.\n");
        $file .= ("     Process_Date: " . $file_date . "\n");
        $file .= ("     Process_Contact:\n");
        $file .= ("       Contact_Information:\n");
        $file .= ("         Contact_Organization_Primary:\n");
        $file .= ("           Contact_Organization:\n");
        $file .= ("             Map and Geographic Information Center,\n");
        $file .= ("             Homer Babbidge Library, Connecticut.\n");
        $file .= ("             University of Connecticut.\n");
        $file .= ("             Storrs, CT.\n");
        $file .= ("         Contact_Address:\n");
        $file .= ("           Address_Type: Mailing and Physical Address\n");
        $file .= ("           Address: Map and Geographic Information Center U-5m\n");
        $file .= ("           Address: Homer Babbidge Library, Connecticut.\n");
        $file .= ("           Address: 369 Fairfield Road.\n");
        $file .= ("           City: Storrs\n");
        $file .= ("           State_or_Province: CT\n");
        $file .= ("           Postal_Code: 06269-1005\n");
        $file .= ("         Contact_Voice_Telephone: (860) 486-4589\n");
        $file .= ("   Process_Step:\n");
        $file .= ("     Process_Description: The center of aerial photograph was visually located on a\n");
        $file .= ("       1:24,000 topographic quadrangle and the point was given a square buffer
based\n");
        $file .= ("       on the scale of the photography.\n");
        $file .= ("     Source_Used_Citation_Abbreviation: None\n");
        $file .= ("     Process_Date: " . $file_date . "\n");
        $file .= ("     Process_Contact:\n");
        $file .= ("       Contact_Information:\n");
        $file .= ("         Contact_Organization_Primary:\n");
        $file .= ("           Contact_Organization:\n");
        $file .= ("             Map and Geographic Information Center,\n");
        $file .= ("             Homer Babbidge Library, Connecticut.\n");
        $file .= ("             University of Connecticut.\n");
        $file .= ("             Storrs, CT.\n");
        $file .= ("         Contact_Address:\n");
        $file .= ("           Address_Type: Mailing and Physical Address\n");
        $file .= ("           Address: Map and Geographic Information Center U-5m\n");
        $file .= ("           Address: Homer Babbidge Library, Connecticut.\n");
        $file .= ("           Address: 369 Fairfield Road.\n");
        $file .= ("           City: Storrs\n");
        $file .= ("           State_or_Province: CT\n");
        $file .= ("           Postal_Code: 06269-1005\n");
        $file .= ("         Contact_Voice_Telephone: (860) 486-4589\n");
        $file .= ("Spatial_Data_Organization_Information:\n");
        $file .= ("  Direct_Spatial_Reference_Method: Raster\n");
        $file .= ("  Raster_Object_Information\n");
        $file .= ("    Raster_Object_Type: Pixel\n");
        $file .= ("    Row_Count: " . $sid_width . "\n");
        $file .= ("    Column_Count: " . $sid_height . "\n");
        $file .= ("Distribution_Information:\n");
        $file .= ("  Distributor:\n");
        $file .= ("    Contact_Information:\n");
        $file .= ("      Contact_Organization_Primary:\n");
        $file .= ("        Contact_Organization: Map and Geographic Information Center\n");
        $file .= ("      Contact_Position: Map Librarian\n");
        $file .= ("      Contact_Address:\n");
        $file .= ("        Address_Type: Mailing and Physical Address\n");
```

```
        $file .= ("       Address: Map and Geographic Information Center U-5m\n");
        $file .= ("       Address: Homer Babbidge Library, Connecticut.\n");
        $file .= ("       Address: 369 Fairfield Road.\n");
        $file .= ("       City: Storrs\n");
        $file .= ("       State_or_Province: CT\n");
        $file .= ("       Postal_Code: 06269-1005\n");
        $file .= ("     Contact_Voice_Telephone: (860) 486-4589\n");
        $file .= ("     Hours_of_Service: 9-12, 1-5 M-F, 18:30-21:30 M-W\n");
        $file .= ("  Distribution_Liability: MAGIC is charged with the distribution of the State's\n");
        $file .= ("    corporate geographic database and, in cooperation with other mapping
organizations,\n");
        $file .= ("    is committed to offering its users accurate, useful, and current information
about\n");
        $file .= ("    the state. Although every effort has been made to ensure the accuracy of
information\n");
        $file .= ("    errors and conditions originating from physical sources used to develop the
corporate\n");
        $file .= ("    database may be reflected in the data supplied. The client must be aware of
data\n");
        $file .= ("    conditions and bear responsibility for the appropriate use of the information
with\n");
        $file .= ("    respect to possible errors, original map scale, collection methodology, currency
of\n");
        $file .= ("    data, and other conditions specific to certain data. MAGIC endorses but does not\n");
        $file .= ("    support secondary distribution of this data.\n");
        $file .= ("  Standard_Order_Process:\n");
        $file .= ("    Digital_Form:\n");
        $file .= ("      Digital_Transfer_Information:\n");
        $file .= ("        Format_Name: SID\n");
        $file .= ("        Format_Version_Date: 1998\n");
        $file .= ("        Format_Specification: None\n");
        $file .= ("      Digital_Transfer_Option:\n");
        $file .= ("        Online_Option:\n");
        $file .= ("          Computer_Contact_Information:\n");
        $file .= ("            Network_Address:\n");
        $file .= ("              Network_Resource_Name: " . $chap_hotlink . "\n");
        $file .= ("  Standard_Order_Process:\n");
        $file .= ("    Digital_Form:\n");
        $file .= ("      Digital_Transfer_Information:\n");
        $file .= ("        Format_Name: TIFF\n");
        $file .= ("        Format_Version_Date: 1998\n");
        $file .= ("        Format_Specification: None\n");
        $file .= ("        Format_Information_Content: Preservation copy of the scanned image.\n");
        $file .= ("        File_Decompression_Technique: No compression applied.\n");
        $file .= ("        Transfer_size: " . $kb . " KB\n");
        $file .= ("      Digital_Transfer_Option:\n");
        $file .= ("        Offline_Option:\n");
        $file .= ("          Offline_Media: CD-ROM\n");
        $file .= ("          Capability_Information: This file is stored on MAGIC Archive CD " . $CDName . "
with CDID " . $CDID . "\n");
        $file .= ("    Fees: None\n");
        $file .= ("Metadata_Reference_Information:\n");
        $file .= ("  Metadata_Date: " . $now);
        $file .= ("  Metadata_Contact:\n");
        $file .= ("    Contact_Information:\n");
        $file .= ("      Contact_Organization_Primary:\n");
        $file .= ("        Contact_Organization: Map and Geographic Information Center\n");
        $file .= ("      Contact_Position: Map Librarian\n");
        $file .= ("      Contact_Address:\n");
        $file .= ("        Address_Type: Mailing and Physical Address\n");
        $file .= ("        Address: Map and Geographic Information Center U-5m\n");
        $file .= ("        Address: Homer Babbidge Library, Connecticut.\n");
        $file .= ("        Address: 369 Fairfield Road.\n");
        $file .= ("        City: Storrs\n");
        $file .= ("        State_or_Province: CT\n");
        $file .= ("        Postal_Code: 06269-1005\n");
```

```
        $file .= ("     Contact_Voice_Telephone: (860) 486-4589\n");
        $file .= ("     Hours_of_Service: 9-12, 1-5 M-F, 18:30-21:30 M-W\n");
        $file .= ("  Metadata_Standard_Name: FGDC Content Standards for Digital Geospatioal
Metadata\n");
        $file .= ("  Metadata_Standard_Version: FGDC-STD-001-1998\n");
        $file .= ("  Metadata_Access_Constraints: None\n");
        $file .= ("  Metadata_Use_Constraints: None\n");

# Write the text files to disk:
        chdir("/magic/working/perlcode/");
        sysopen (TMP, "text_files/" . "$file_name" . ".txt", O_RDWR | O_CREAT) or die "can't open file:
$!";
        print TMP $file or die "can't write file: $!";
        close (TMP) or die "can't close file: $!";
        }

}
############################
### file_process     ###
############################
# This subroutine runs the mp command that produces the .xml, .sgm,
# .htm, and text files. The .err, and .dif files are no longer being created.
sub file_process {
        print "Beginning mp command...\n";
        chdir "text_files" or die "cannot chdir to text_files: $!";
        $sth4 = $dbh->prepare("SELECT file_name FROM meta");
        $sth4->execute();
        $sth4->bind_col( 1, \$file_name );
        while ($sth4->fetchrow_array()) {

# old method
#       system("../mp.sun -x " . $file_name . ".xml  -s " . $file_name . ".sgm -h " . #$file_name . ".htm
-e " . $file_name . ".err -d " . $file_name . ".dif " . $file_name . #".txt");

#new method
        system("../mp.sun -x " . $file_name . ".xml -s " . $file_name . ".sgm -h " . $file_name . ".htm " .
$file_name .".txt");
        }
        $sth4->finish;
        print "Finished creating xml,sgm,htm,err and dif files!\n";
}
############################
### fix_chapdate     ###
############################
# Subroutine to properly format the chapdate for the mp program.
sub fix_chapdate {
        my $string = $_[0];
        my $year = substr($string, 0, 4);
        my $month = substr($string, 4, 2);
        my $day = substr($string, 6, 2);
                $month =
          ($month eq "01") ? "January"  :
          ($month eq "02") ? "February" :
          ($month eq "03") ? "March" :
          ($month eq "04") ? "April" :
          ($month eq "05") ? "May" :
          ($month eq "06") ? "June" :
          ($month eq "07") ? "July" :
          ($month eq "08") ? "August" :
          ($month eq "09") ? "September" :
          ($month eq "10") ? "October" :
          ($month eq "11") ? "November" :
          ($month eq "12") ? "December" :
                     "December"; # default
        my $long_date = $month . " " . $day . ", " . $year;
        return $long_date;
```

```
}
#########################
### fix_filedate      ###
#########################
# Subroutine to properly format filedate.
sub fix_filedate {
        my $string = $_[0];
        my ($day, $month, $year) = split(/-/,$string,3);
                $month =
          ($month eq "Jan") ? "01"  :
          ($month eq "Feb") ? "02" :
          ($month eq "Mar") ? "03" :
          ($month eq "Apr") ? "04" :
          ($month eq "May") ? "05" :
          ($month eq "Jun") ? "06" :
          ($month eq "July") ? "07" :
          ($month eq "Aug") ? "08" :
          ($month eq "Sep") ? "09" :
          ($month eq "Oct") ? "10" :
          ($month eq "Nov") ? "11" :
          ($month eq "Dec") ? "12" :
                      "12"; # default
        my $reformed_date = $year . $month . $day;
        return $reformed_date;
}
```