PROGRAMMING AN IMAGE DATABASE MANAGEMENT SYSTEM FOR USE INSIDE A CAD-ENVIRONMENT

Claus Lichtenberg, Dipl.-Ing.
Guenter Pomaska, Prof. Dr.-Ing.
FH Bielefeld, University of Applied Sciences
Faculty of Architecture & Civil Engineering
Artilleriestrasse 9, D - 32427 Minden, Germany
E-mail: gp@imagefact.com

Comission V, Working Group V/2

KEY WORDS: Image database, Managing photogrammetric images, Database management foundations, CAD

ABSTRACT

This article explains the basic concepts of developing an image database to use inside a CAD-Environment. Beside general techniques and requirements on a database, special requirements raised from photogrammetric images will be discussed. Further more we will introduce the image database and the way it work's.

Der Beitrag behandelt grundlegende Konzepte zur Entwicklung einer Bilddatenbank innerhalb einer CAD-Umgebung. Neben der Darstellung genereller Anforderungen und Techniken der Datenbankentwicklung, werden auch die sich bei der Verwaltung photogrammetrischer Bilddaten ergebenden Besonderheiten diskutiert. Darüber hinaus wird die Funktionsweise der Bilddatenbank vorgestellt.

1. PREFACE

The permanent developement in photogrammetric technologie and the increasing trend to digital image analysis presume high qualitiy computer hardware. In case of permanently decreasing costs for computer hardware you find more and more efficient workstations which are able to solve complex photgrammetric requirements. Getting high qualitiy images with extensive data information is one of the major aspects by planing a photogrammetric project. In general the backup and the mangement of the images and their data achieves only a few consideration.

In some cases this might be okay because in general the images exist as a photo or slide and the corresponding data can be rebuild with a program. This point of view might be correct in case of small projects with only a few pictures and data. In greater projects the reconstruction and analysis of the images and their data often tend to take a lot of time and thus costs.

Wouldn't it be nice under this circumstances to have a tool that offers the ability to store images with their data so that you have easy access to the requiered information at any time?

The image database tries to attend this claim. The programming language used to develope the image database was Borland Delphi 2.0. For the database management we choosed the Paradox format because it's part of the Delphi package. Another reason for the decision on Paradox was that this database concept was originally introduced by Borland and so the Delphi package offers a native database driver with good performance. For the CAD environment we selected AutoCAD Realease 13 from Autodesk. The fundamentals of the image database and the way it works will be discussed further on.

2. REQUIREMENTS ON PROGRAMMING AN IMAGE DATABASE

2.1 General Requirements

Before you start implementing a database you have to have a close look on the data and their structures which should be managed. Furthermore you have to keep in mind the outfit and appearance of the program.

While the outfit and appearance of the program is determined by general requirements the data storing and database conception is determined by special requirements which will be discussed later on.

The general requirements are defined as:
- easy to use handling

- Windows 95 outlook
- Different access levels to protect data
- Tools and utilities to handle data and information

The major role of the Windows 95 user platform was the main reason for developing the image database as a Windows 95 conform program. Because of the embedded 32-Bit features in Borland Delphi the image database can also be used under Windows NT 4.0. This is important because in commercial offices and network environments you often find the Windows NT 4.0.platform.

With the release 4.0 of Windows NT Microsoft uses the same workplace outfit under this user platform as known before from Windows 95. So the requirement of a Windows 95 outlook or so called look and feel is a logical consequence.

To provide the data of the image database from illegal access and manipulation, different user access levels were necessary.

Beside general items to handle data the user needs for his convinience special tools and utilities. For example: if the user wants further information on a special image, the program has to offer him a tool to achieve this informations easily. Furthermore he often need's utilities. For example when he need's to recreate a table or export datasets to different programs.

The way the software developer implements those general requirements decides over the qualitiy of a program and the advantage for the user. Even if there are a lot of rapid application development tools on the market which offer the developer to concentrate on the special requirements of a program, you often find so called "banana software" which is a synonym for software that mature by the customer through endless patches and fix packs.

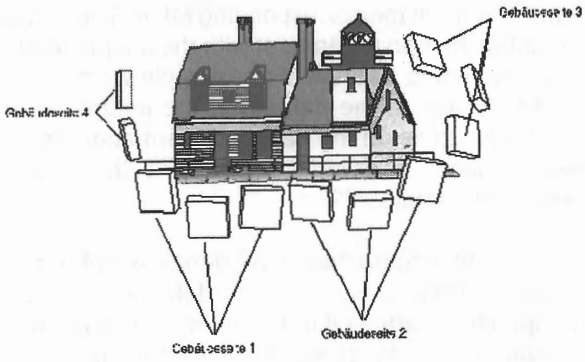

Figure 1

2.2 Special Requirements

Before you can handle data you have to analyze the process of getting data and the relationships between data of different kinds.

When analyzing the process of taking photogrammetric images we found out that in general it follows an hierarchical order.

For example: If you take photogrammteric pictures from a building a group of pictures represents a special part of the building. The complete set of pictures make up the building itself. If the building is part of a street which you have to analyze, then you get the hierarchical order shown below.

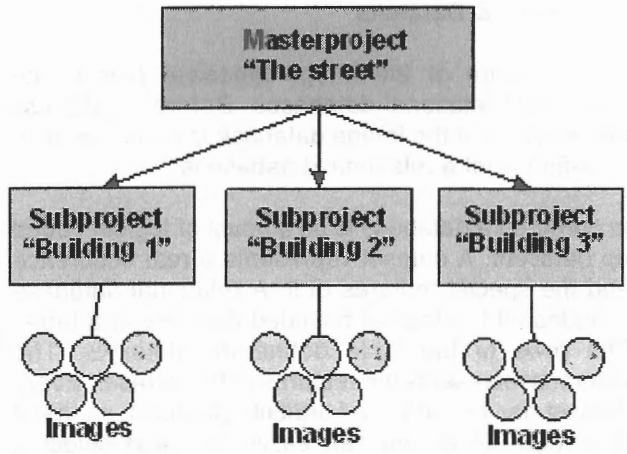

Figure 2

This hierarchical order determines the structure of the image database and it's tables. Furthermore it defines a special requirement because the image database program has to take care of this structure when it presents the data to the user. This means the user interface has to represent this structure in a hierarchical way.

Another special requirement raised from the pretension to use the image database under CAD. The goal was to reference the images of the database by a camera symbol inside the CAD environment. If the user selects a camera symbol inside the drawing he can select the corresponding image by it's image number from the database. So he has the possibility to check the camera position inside his CAD drawing. Two major claims raise from this procedure. First of all the image number which corresponds to the camera symbol must be unique. Second, the camera position co-ordinates and the accuracy of this co-ordinates (approximate values, bundle adjustment, single image orientation) must be part of the image dataset.

One more special requirement result's from visualization. If the user want's to map an image on a CAD model he needs the image coordinates from the

lower left corner and the upper right corner of the image to fit it on the model. So the database has to store these data as well in an image dataset. In this coherence a tool to apply a coordinate offset is also important.

Last but not least there has to be a possibility to transfer data from the image database to other applications especially to AutoCAD and to import data to the image database for example from the Rolleimetric CDW program.

3. FUNDAMENTALS OF RELATIONAL DATABASES

3.1 Tables & Datasets

The structure of the image database follows the concept of relational databases. Before we discuss the structure of the image database in detail we have to define what a relational database is.

In common a database is an amount of logical bound up datasets. A dataset represents a real occurence and the special features of it. A relational database includes all the logical bounded datasets in a table. The rows of the table define the datasets. The columns represents the features of the dataset. Every dataset has an individual attribute (feature), so that it is unique. Thus you can easily find and select a dataset from the table. The unique feature is called "primary key". The dependencies between rows, columns, datasets and attributes are illustrated in figure 3.

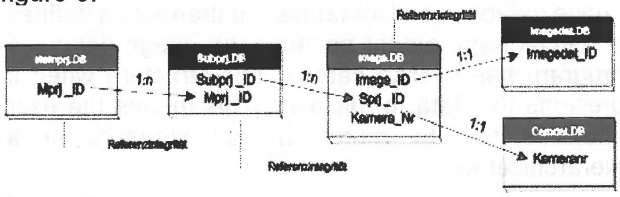

Figure 3

Img_ID
Img_No.
Img_Res
Img_file
0001
1005
640x480
tif
002
1006
640x480
Gif
...
...
...
...

3.2 References & Relations

In the image database we have to store different types of datasets. As discussed before we have to represent the hierarchical structure of photogrammetric projects. We did this by defining two tables one for the so called "Masterproject" the other for the "subproject". The Masterproject table and the subproject table are very similar. Both tables contains datasets which specify the project by a title, a date and comment. As seen before a subproject is part of a Masterproject. So there is a dependency between the two tables. The relationship is a 1:n - relation. This means one dataset in the Master-project table may have n corresponding datasets in the subproject table.

The subprojects are build up by the images which belongs to one special subproject. So the we defined a "Image-table" which has also a 1:n - relation to the subproject table.

Beside the camera standpoint coordinates we have to store the image coordinates and further informations to the image itself. Information on the images are stored in the above explained "Image-table". The image coordinates were stored in the "Image data table". Each image only has one corresponding dataset in the image data table. This is called a 1:1 - relation.

Finally we have the camera data to be stored in a table. The camera data depence on the type of camera you used to get your photogrammetric images. We stored this data in the "camera table". An image may have one corresponding camera dataset.

In addition to the explainations of the dependencies we have to discuss how they can be realized. As we saw before each dataset has a unique feature. To connect datasets from different tables we use this unique feature in the corresponding table. This means we expand the master dataset with the unique feature of the depending dataset. For example we store the camera number in the dataset of the image. So we can easily access on the camera information for one special image. But what happens if there is no corresponding dataset?

To avoid this circumstance we define a "referentiell integrity". This means if a dataset needs a corresponding dataset it only can be stored when the corresponding dataset exists. This technique also prevents the database from corruption when deleting a dataset but not the corresponding ones. For example: Before deleting a Masterproject you have to delete all subprojects which belong to this Masterproject. The subproject itself only can be deleted when all images that correspond to this subproject were deleted and so on. Figure 4 shows the relations between the tables of the image database and their referentiell integrities.

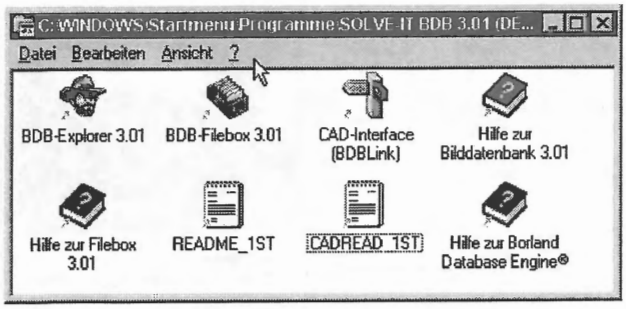

Figure 4

3.3 Questions & Answers

A great advantage of a data based application is it's ability to get the informations the user wants to have. To create those information sets you need a tool that allows you to generate specific queries on the datasets. This tool is called "structured query language (SQL)". The structured query language uses basic keywords. By the combination of those keywords you can create complex queries on the database. A possible query on the image database could be: "Show me all image numbers from subproject x, where the data quality fits to bundle adjustment." As a SQL statement you get: "SELECT * FROM Image.db, subproject.db WHERE img_qual = bundle_adjustement"

The introduced relational database model guarantees consistent and non redundant datasets. Beside the relational database structure you need a database language to achieve access on datasets, result sets and selection sets. The most spreaded database language is the "Structured Query Language (SQL) ". SQL has achieved a widely accepted industry standard.

The SQL language contains instructions for data definition, data selection, data manipulation and data administration. Every access on datasets follows a quantity orientated design at what all operations from relational algebra can be used. Especially on queries about related datasets SQL offer a mighty tool to get the special information.

For example: If you would like to filter all the images which belong to one special subproject (ID = 2)the SQL statement be issued to:
"SELECT * FROM Image.DB
WHERE SubprjID = 2
The "select instruction" used in this sample always creates a result set which can be used for further SQL statements. SQL statements which refer to other SQL statements are called "sub queries. By using logical operators in the "where" instruction part of the statement you can filter datasets which attributes have to fit in more than just one feature. According to the statement above a possible query could be: "Show me all the images from subproject 2 where the author was Mister A or Mister B".
SELECT * FROM Image.DB
WHERE SubprjID = 2
AND Autor = 'Mister A'
OR Autor = 'Mister B'
There are much more features about SQL. To discuss all of them would exceed this excurse. For further information on SQL a wide range of literature is available.

4. THE IMAGE DATABASE

The image database consists of four different applications. The BDB-Explorer, the BDB-Browser (part of the BDB-Explorer), the BDB-Filebox and the CAD-interface. The following paragraphs will introduce these applications.

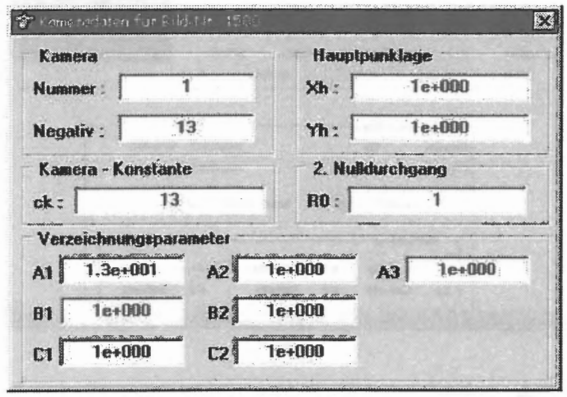

Figure 5

4.1 The BDB-Explorer

The BDB-explorer is the main program of the image database. It offers the user all tools and routines to manage and handle the data of the image database. Because of the hierarchical order the BDB-explorer follows in it´s design the concept of the well known Windows 95 explorer.

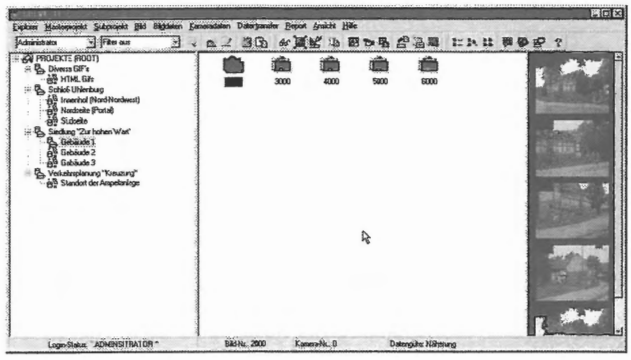

Figure 6

The treeview of the BDB-explorer shows the master- and it´s corresponding subprojects, the listview contains the images which belong to one subproject.

On the right hand you see the "image panel" which gives the user a quick overview over the stored images. Different viewstyles and plenty of information dialogs offers the user all the information he needs to manage his projects and data.

Furthermore there are standard information dialogs implemetented. The "Quick-Info" dialog shown in figure 7. contains basic informations on the selected image.

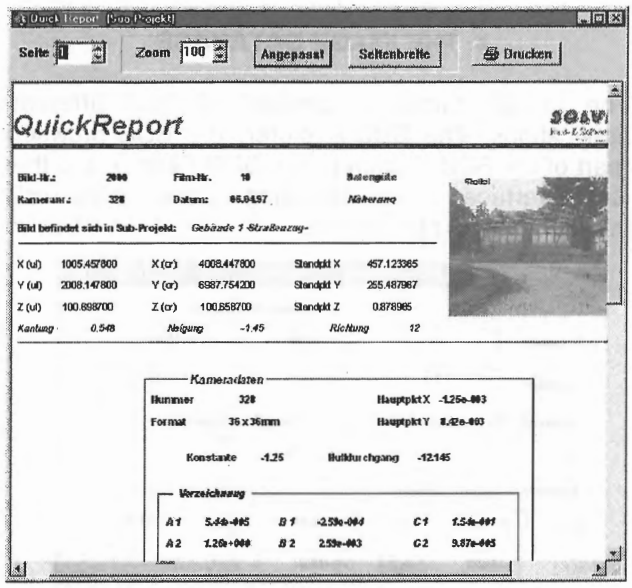

Figure 7

The image dataset dialog shown in figure 8 contains informations about the image (number, camera, date, author) as well as image coordinates and detailed

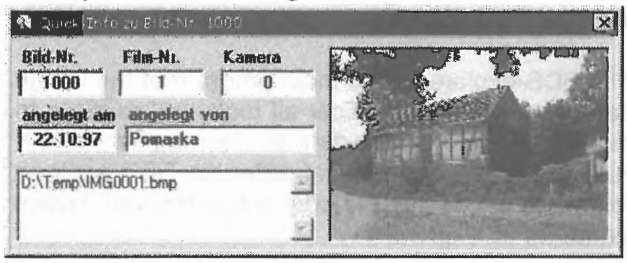

Figure 8

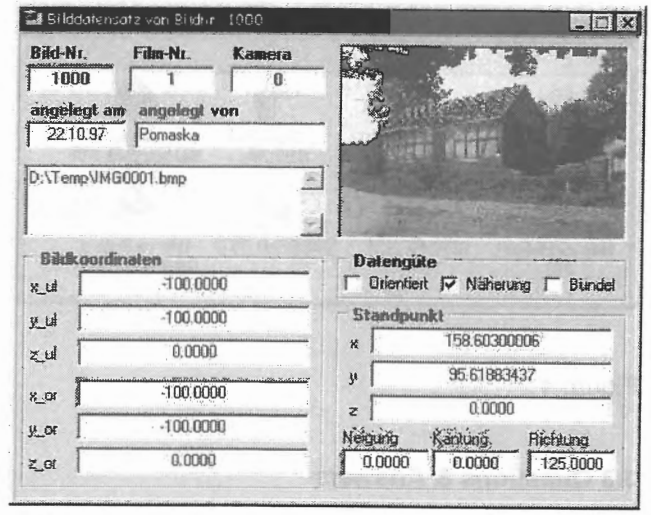

Figure 9

The camera information dialog offers the user all relevant parameters of the corresponding camera.

To bring the received information on paper an extensive report was implemented.

Beside the information and report dialogs all database management routines are implemented in the BDB-Explorer. So it is very important to control the level a user has access on them. The BDB-Explorer takes care of this by using different user access levels. The "guest-level" only allows the user to have access on basic data informations. Deleting or modifying data is not possible. The "user-level" was designed for data maintenance. The "administrator-level" offers full access to all data management tools and routines.

4.2 The BDB-Browser

The BDB-Browser was designed for the professional power user. With the BDB-Browser the user has a mighty tool to his disposal. The BDB-Browser enables the user to handle the complete database management. This means, the user can work on each setting of the database. He can change datasets, data definition, keys etc. Furthermore he can manage the database connection and the access rights on a system level. Last but not least the BDB-Browser offers tools for data maintenance like packing, copying or converting a table and it's data.

Because incautious handling can destroy the database integrity and thus the functionality of the entire image database it strongly recommended that only advanced users use the BDB-Browser. To guarantee this the BDB-Browser is only avaiable on the "administrator level".

4.3 The CAD-Interface

AutoCAD Release 13 for Windows contains an own

database connector module. This module requires a full functioned database management system. The connection procedure between datasets and drawing elements as well as the creation of result datasets is especially for the non professional user not easy to handle. Because of this fact we tried another way to connect the image database to the AutoCAD environment.

To enable the user to have easy access on the images and their data inside the CAD environment was one major goal when developing the CAD-Interface. Beside the easy access on data and informations, routines to exchange data between CAD and the image database were required. The developed CAD-Interface builds up the connection to the CAD environment. It offers the user the possibility to have access on the images and their data. To connect the image database while working in a CAD environment you have to start the application called "CAD-interface". After it starts up you have to select the Master- and subproject on which data you like to have accees to. After this procedure the so called "Image Browser panel" appears. For every image of the subproject there is a camera symbol associated with an image number. On figure 10 you can see the image browser on the right side.

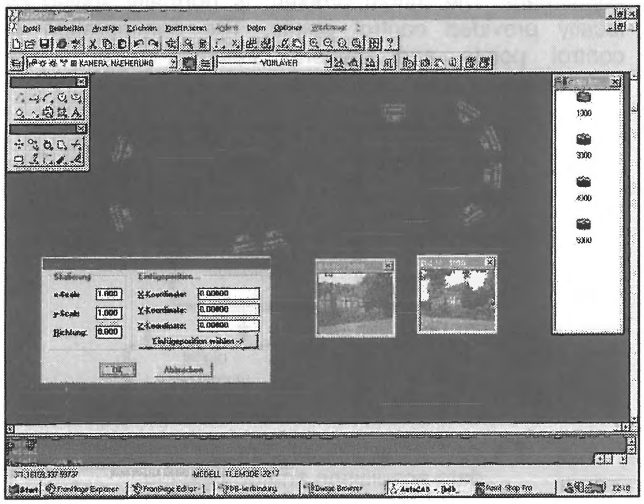

Figure 10

When you click on a camera item inside the panel a pop-up menu appears and offers you different options like quick view, quick-info, an image data dialog or a camera data dialog. The image browser panel itself and the data access routines were implemented as dynamic link libraries (dll's). This was neccessary because the browser panel and the corresponding dialog boxes always has to stay on top of screen even when the CAD application is activated by the user.

As discussed above the reference inside the CAD drawing is a camera symbol. To place the camera symbol which references an image inside your CAD drawing we designed a block with extended attributes (camblk.dwg). These attributes are the image number and the data quality. For a comfortable use we developed some LISP-routines which helps the user to place and handle the camera blocks.

4.3.1 Direct data exchange

By opening the corresponding image data dialog the user can change the coordinates by typing in the new coordinates. When the dialog is closed the data will be updated and stored automatically in the image database.

4.3.2 Clipboard data exchange

By using the LISP-Rouine "Caminfo" a dialog box appears which contains the actual CAD coordinates of the camera block. Copy these coordinates to the clipboard and insert them in the appropriate coordinate dialog box of the image. When the image coordinate dialog box is closed the data will be automatically updated.

4.3.3 Transfer file data exchange

The data transfer file to export data will be created with the AutoCAD block extraction routine. The required pattern file is part of the image database. By extracting the camera block attributes it is very important to create a space delimited file (SDF-file format). If neccessary you can modify the pattern file for your needs.

To import data from the image database into the CAD drawing you can create a CAD transfer file. With this file the camera symbols will be automatically placed in your CAD drawing at the appropriate position. To read in the transfer file you use the LISP routine called "Caminp.lsp". The structure of the input file is listed below.

5. Resume

With this article we tried to introduce the image database and it's concepts. We know that it is not easy to imagine how the image database works in practice when only reading about it. To receive an impression about the advantages the image database has to offer you in practice .You are welcom to visit our website under **http://www.imagefact.com**. Here you can download a free demoversion of the image database.