

XML basierte Datenformulierung zur Web-konformen Dokumentation photogrammetrischer Bauaufnahmen

GÜNTER POMASKA, Bielefeld & NIKOLAI DEMENTIEV, Wologda/Russland

Keywords: photogrammetry, archiving, CAD, Cultural Heritage, close range, Internet/Web

Einleitung: Mitte der achtziger Jahre wurden photogrammetrische Messsysteme in den Markt eingeführt, die nach dem Verfahren der Mehrbild-photogrammetrie auf PC-Systemen lauffähig waren. Vergrößerte Papierabzüge der Messbilder waren auf einem Digitizer punktweise auszumessen. Nachteilig wirkte sich der Arbeitsaufwand bei der Orientierung der Messbilder auf dem Digitizer aus. Unvermeidbar der hohe Kostenfaktor, der durch den notwendigen Einsatz einer Réseaukamera entstand.

Durch fortschreitende Technologie der digitalen Bildverarbeitung entwickelten sich die Systeme weiter. Mit Notebook und kalibrierter digitaler Spiegelreflexkamera sind jetzt wirtschaftlich einsetzbare Photogrammetrie-Systeme für den Arbeitsplatz verfügbar, die sich auch nahtlos in CAD-Arbeitsumgebungen einfügen.

Photogrammetrie liefert die Grundlage zur Datenaufbereitung aus einem „dreidimensionalen Bild“, zu vergleichen der beim Laserscanning erzeugten Punktwolke. Abzuwägen ist, ob die vollständige Auswertung der Dienstleistung eines Vermessungsfachmanns zu übertragen ist, oder ob die Weiterverarbeitung in einem auswertbaren Rechnermodell bei Bedarf erfolgen kann.

Der vorliegende Beitrag betrachtet die Möglichkeiten einer Web-konformen Formulierung aller bei photogrammetrischen Bauaufnahmen anfallenden Daten. Dabei liegt das Augenmerk auf der allgemein zugänglichen Ergebnisdokumentation und auf dem Zugriff auf die Ursprungsdaten zur Bestimmung weiterer Größen von verteilten Arbeitsplätzen, die mit Internetzugang ausgestattet sind.

Introduction: *XML based data description for Web compliant photogrammetric documentation of buildings.* Photogrammetric desktop systems, operating with the multi image principle, were introduced in the mid eighties. Enlarged prints of the images had to be surveyed on digitizers point by point. Two major disadvantages occurred. Orientation of the photos on the digitizer was time consuming. The request for a réseau-camera increased the investment.

Concurrently with the ongoing technology of digital image processing, the systems became more and more usable. Notebook computer and calibrated digital SLR camera are the hardware components of a cost-effective photogrammetric system, fitting very well in a CAD environment today.

Photogrammetry provides a 3D-image, comparable with a point cloud from laser scanning. From the raw data, 3D measurements can be taken upon request.

This contribution reviews the possibilities of a Web compliant description of a complete photogrammetric data set. Taking into consideration the access and the further evaluation with distributed workstations connected to the Internet.

1 Die Zitadelle in Wesel

Unter Friedrich Wilhelm I., König in Preussen – Regierungszeit 1713–1740 –, erreichte die Festung Wesel im 18. Jahrhundert ihre größte Ausbaustufe. Aus dieser Zeit datieren in der Zitadelle: Das Haupttor mit Toranlage (bis 1718) sowie die Kaserne VI (1727/28) und in der Stadt: Die Lutherkirche (1729) und das Berliner Tor (1718–1722), sowie aus der Zeit der ersten beiden preußischen Könige Reste der Niederflanke der Bastion Friedrich-Wilhelm (zw. 1702–1727). Das Haupttorgebäude besitzt eine auffällige Verwandtschaft mit dem um 1698 entstandenen Entwurf Jean de Bodts eines Invalidenhauses für König Wilhelm IV. von England. Die Risalite der Außenfronten entsprechen hier in Proportionen und Gliederungen in etwa den Flügeln des Haupttorgebäudes, die sich zum Zitadellenhof öffnen. Dies gilt auch für die Mansardendächer, die am Haupttorgebäude der Zitadelle nach dem 2. Weltkrieg nicht mehr aufgesetzt wurden.

Da Jean de Bodt seit 1703 Arbeiten in Wesel verrichtete, dürfte ein deutlich früheres Entstehungsdatum als 1718 durchaus in Be-

tracht kommen. Das Haupttorgebäude diente als Wachgebäude, Gefängnis, zeitweise auch als Kommandantur. Die spätere Kaserne VII wurde als Kornmagazin erbaut, 1763 zur Kaserne umgestaltet. Das Haupttorgebäude der Zitadelle ist entweder 1718 zeitgleich mit dem Außentor entstanden oder bereits älter.

Der dem Zitadellenhof zugewandte Mittelrisalit in Sandstein besitzt mit seinen gekröpften Pilastern und dem gleichfalls gekröpften Giebel starke Bezüge zu Manierismus und Frühbarock. Viel später als 1700 dürfte dieser Bau nicht errichtet worden sein. Die Urheberschaft von Jean de Bodt an diesem Bauwerk ist sehr wahrscheinlich.

2 Aufnahmesysteme, Kalibrierung und Objektaufnahme

Die photogrammetrische Aufnahme erfolgte nach dem Verfahren der Mehrbildphotogrammetrie, gestützt von einigen terrestrisch bestimmten Kontrollpunkten. Zum Vergleich wurden die Aufnahmesysteme Rolleiflex metric 6008, Mittelformat, Farbnegativfilm, 50 mm Brennweite und eine di-

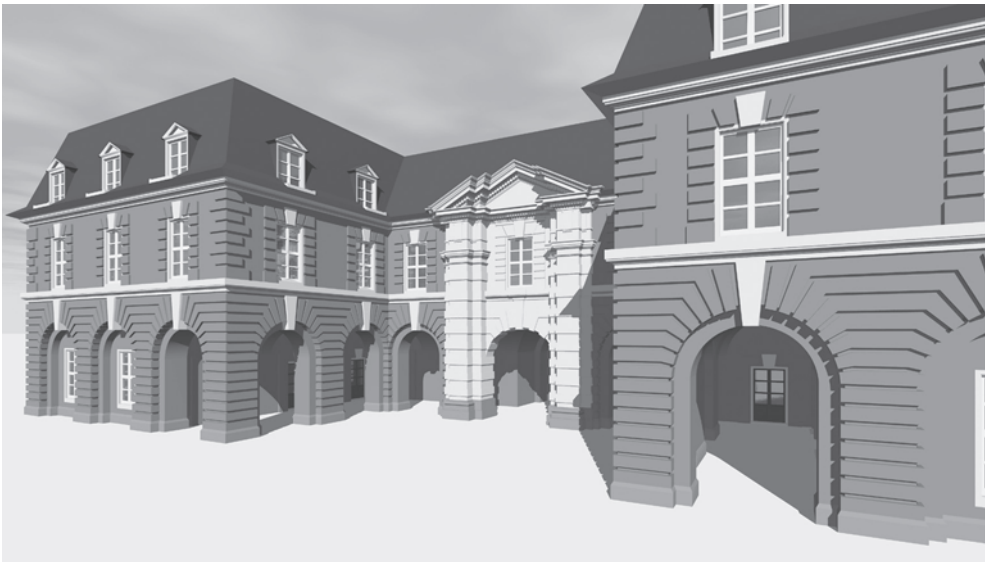


Abb. 1: Bauaufnahme der Zitadelle in Wesel: 3D-Modell mit Rekonstruktion der Mansardendächer vor 1945.

digitale Kleinbildkamera Nikon D70 eingesetzt. Die Sensorfläche der D70 hat eine Größe von $23,462\text{ mm} \times 15,600\text{ mm}$. Bei einer Auflösung von 3008×2000 Pixel beträgt die Pixelgröße 7,8 Mikrometer. Als Aufnahmeobjektiv wurde das 20 mm F1,8 EX DG Aspherical RF von Sigma eingesetzt. Technische Details zum Objektiv sind auf der Webseite <http://www.sigma-foto.de> spezifiziert.

Kalibrierdaten für das digitale Aufnahmesystem wurden von der Firma AICON GmbH, Braunschweig, freundlicherweise zur Verfügung gestellt. Die Abnahme eines photogrammetrischen Systems erfolgt heute i.d.R. mithilfe eines in der VDI/VDE Richtlinie Nr. 2634 Blatt 1 „Optische 3D Messsysteme, Systeme mit flächenhafter Antastung“ beschriebenen Kalibrierkörpers. Der Kalibrierkörper von AICON enthält 58 DKD-kalibrierte Teilstrecken. Die dreidimensionale Längenmessabweichung dL ergibt sich aus der Differenz zwischen dem gemessenen Wert und dem kalibrierten Wert eines Abstandes zweier Punkte (Teilstrecke). Der Grenzwert E der zulässigen dreidimensionalen Längenmessabweichung ist die Kenngröße für die Längenmessabweichung.

Zur Ermittlung dieser Daten wird mit dem Aufnahmesystem ein Bildverband von ca. 50–100 Bildern des Kalibrierkörpers aufgenommen. Das für die Architekturphotogrammetrie relevante Ergebnis einer Ka-

librierung ist ein Datensatz mit den wesentlichen Parametern der inneren Orientierung (Kammerkonstante, Lage des Bildhauptpunktes im Bildkoordinatensystem, Verzerrungsparameter).

Die photogrammetrische Aufnahme des Haupttorgebäudes der Zitadelle wurde in mehrere Bildverbände mit jeweils konvergenten Aufnahmeanordnungen unterteilt. Einzelne Bildverbände bestehen aus etwa 6 bis 8 Messbildern. Bildmaßstäbe liegen im Bereich zwischen 1:100 und 1:400. Im vorliegenden Anwendungsfall ist die Segmentierung der Gesamtaufnahme in einzelne Bildverbände mit geringer Anzahl von Messbildern eine bewährte Methode. Mit den vorgegebenen Kalibrierwerten wurde eine Messgenauigkeit bei der Bündelausgleichung eines Bildverbandes von 3 Mikrometer erzielt. Diese hohe Messgenauigkeit bei der Messung von natürlichen Punkten wurde nicht erwartet. Im Vergleich mit dem analogen Aufnahmesystem Rolleiflex 6008 metric ist zu berücksichtigen, dass für dieses System die werkseitige Kalibrierung nicht in Kombination mit dem Digitalisierungssystem, Agfa Filmscanner, vorliegt. Das Ergebnis der Bündelausgleichung lag mit letzterem System in der Größenordnung um 15 Mikrometer; aufgrund des größeren Bildmaßstabes ist aber gegenüber dem Vergleichssystem kein signifikanter Verlust an Objektgenauigkeit zu verzeichnen. Die Bestimmung der Detailpunkte am Objekt erfolgte mit beiden Aufnahmesystemen mit einer Genauigkeit von einem Zentimeter. Das entspricht der Identifizierbarkeit am Objekt und ist für die Aufgabenstellung ein wirtschaftlich erzielbarer Wert.

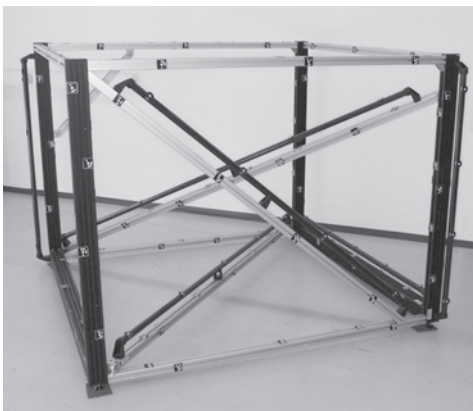


Abb. 2: AICON Kalibrierkörper.

3 Datenformulierung in XML Extensible Markup Language

XML (Extensible Markup Language) ist eine Metasprache zur Strukturierung von Daten. XML verwendet Tags, Schlüsselbegriffe in spitzen Klammern $< >$ und Attribute. Gegenüber HTML ist die Bedeutung der Tags und Attribute in XML nicht definiert und wird erst durch die Anwendung interpretiert.

XML-Dateien sind Textdateien, zu deren Darstellung weitere Technologien notwendig sind. Mit der Extensible Stylesheet Language (XSL) besteht die Möglichkeit, XML-Dateien für Webbrowser formatiert aufzubereiten oder in Druckformate wie PDF zu konvertieren. Wir werden zunächst die photogrammetrischen Daten unseres Architekturprojekts in XML formulieren und hier in Auszügen wiedergeben.

Die Daten eines photogrammetrischen Bildverbandes, die für die weitere Bearbeitung relevant sind, umfassen die Kameradaten, die Fotopositionen, tachymetrisch bestimmte Kontrollpunkte, korrespondierende Objektpunkte der Bündelausgleichung und photogrammetrisch bestimmte grafische Elemente der Detailauswertung.

XML-Dokumente beginnen mit einem Prolog, das ist der Bereich vor dem Wurzelement, der hauptsächlich die Dokumententypdefinition und Angaben zur DTD (Document Type Definition) enthält. Im weiteren Verlauf der Betrachtungen kommen wir auf die DTD zurück.

Zunächst wird das XML-Dokument wie folgt notiert:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<!DOCTYPE imageBundle SYSTEM "imageBundle.dtd">
<imageBundle>
  <!-- Inhalt des Dokuments -->
</imageBundle>
```

Mit dem Wurzelement *imageBundle* wird eine Instanz der Dokumentenklasse, die in der !DOCTYPE Anweisung angegeben ist, angelegt. Ein Element mit Unterelementen wird durch einen Knoten im Strukturbaum des XML-Dokuments dargestellt. Eine Kamera besteht aus den Elementen: Bezeichnung, Kammerkonstante, Hauptpunktlage, Verzeichnung, Bildformat. Ein Fotostandpunkt wird formuliert durch die Bezeichnung, benutzte Kamera, Koordinaten des Aufnahmeorts und Richtung der Aufnahme. Start-Tag und Ende-Tag einer XML-Datei müssen mit Ausnahme des vorangestellten Schrägstrichs im Ende-Tag genau übereinstimmen. Ein Element besteht aus

Start-Tag, Inhalt (Wert) und Ende-Tag. Attribute sind innerhalb des Start-Tags paarweise in der Kombination Name und Wertzuweisung abgelegt. Die Daten der Aufnahmesysteme sind in der XML-Struktur wie folgt gespeichert:

```
<cameraData>
  <camera>
    <type>nikon-28</type>
    <ck>-18.23718</ck>
    <xh>-0.09973</xh>
    <yh>-0.01304</yh>
    <a1>-3.03846E-004</a1>
    <a2>6.43569E-007</a2>
    <formX>23.462</formX>
    <formY>15.600</formY>
  </camera>
</cameraData>
```

Betrachtet man eine XML-Datei mit einem Web-Browser, dann zeigt dieser die Baumstruktur des Dokuments an. Die einzelnen Knoten können durch Klick auf das vorangestellte Zeichen geöffnet oder geschlossen werden. Im MSIE stellt sich der Strukturbaum wie in der Abb. 3 dar. Zur formatierten Betrachtung im Browser bedarf es einer

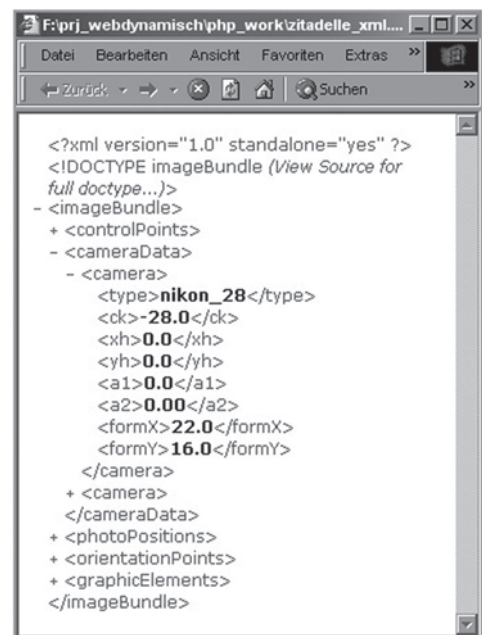


Abb. 3: Strukturbaum einer XML-Datei, angezeigt im MSIE.

Transformation. Die Umsetzung erfolgt mit XSL (Extensible Stylesheet Language). Der Einsatz des XSL-Prozessors kann serverseitig, online im Browser oder offline erfolgen. Im letzteren Fall wird dann die HTML-Datei auf dem Server vorgehalten. Als weiteres XML-Werkzeug findet XPATH Anwendung. Mit XPATH besteht die Möglichkeit, Auszüge aus einer XML-Datei zu erstellen und nach Mustern zu durchsuchen. Wir referenzieren die externe XSL-Definition im Prolog der XML-Datei wie folgt:

```
<?xml-stylesheet          version="1.0"
href="template.xsl"      type="text/
xsl"?>
```

Ein Ausschnitt aus der Stilvorlage *template.xsl* zeigt die Verarbeitungsinformationen für die Ausgabe der Kameradaten in einer HTML-Tabelle. Innerhalb einer `<for-`

`each></for-each>` Anweisung wird nach dem Muster *imageBundle/cameraData/camera* gesucht und mit `value-of select` das gewünschte Element ausgewählt. Das Präfix `xsl` definiert den Namensraum, dem die Anweisung zuzuordnen ist. Der Tag `<tr></tr>` ist ein HTML-Tag zur Definition einer Tabellenzeile, `<td></td>` definiert den Inhalt einer Tabellenzelle. Das Attribut `class` mit dem Wert *tab_value* referenziert die Vorgaben des Layouts, die mit Cascading Style Sheets (CSS) innerhalb des XSL-Dokuments im Tag `<style></style>` eingetragen sind.

```
<xsl:for-each select="imageBundle/
cameraData/camera">
  <tr>
    <td class = "tab-value"><xsl:va-
lue-of select="type"/></td>
```

Photogrammetrische Bauaufnahme Zitadelle Wesel [FH BI 10.2004]								
Kontrollpunkte / Tachymeteraufnahme:								
Punkt.Nr	code	x	y	z				
01	99	212.761	83.730	11.164				
02	99	200.000	100.000	10.000				
02	99	201.346	100.007	10.009				
Aufnahmesysteme:								
Type	ck	xh	yh	a1	a2	Format b/h		
nikon_28	-28.0	0.0	0.0	-.5	-0.05			
6008_50	-51.024	-0.25	-0.16	-1.07400E-005	1.04300E-002			
Photo-Positionen:								
Bezeichnung	Kamera-Typ	x0	y0	z0	wx	wy	wz	Bilddatei href
Orientierungspunkte:								
PNr	code	x	y	z	sx	sy	sz	
35	88	212.006	99.960	13.686	0.003	0.006	0.003	
37	88	207.409	99.949	13.680	0.003	0.006	0.003	
42	88	212.301	100.045	17.953	0.004	0.009	0.005	

Abb. 4: XSL-Transformation der XML-Datenstruktur in eine HTML-Tabelle.

```

<td class = "tab-value"><xsl:va-
  lue-of select="ck"/></td>
<td class = "tab-value"><xsl:va-
  lue-of select="xh"/></td>
<td class = "tab-value"><xsl:va-
  lue-of select="yh"/></td>
</tr>
</xsl:for-each>

```

XML-Dokumente können wohlgeformt oder gültig sein. Aus einem wohlgeformten XML-Dokument wird durch Ergänzung mit einer Document Type Definition (DTD) ein gültiges Dokument. Die DTD enthält das Vokabular und die Regeln zur Verarbeitung der Informationen eines XML-Dokuments. Definiert werden alle Elemente, Attribute und Entitäten und Angaben zu Anzahl, Inhalt und Verschachtelung der Elemente. Die DTD wird mit der Anweisung !DOCTYPE im Prolog eingeleitet. Die Definition selbst, kann eingebettet oder extern in einer Datei gespeichert sein. Ein Auszug aus der DTD für den Bereich der Kameradaten, der auf die Kontrollpunkte folgt, gestaltet sich wie folgt:

```

<!DOCTYPE imageBundle [
  <!ELEMENT imageBundle
    (controlPoints*,
     cameraData*,
     photoPositions*,
     orientationPoints*,
     graphicElements*)>
  <!ELEMENT controlPoints
    (point*)>
  <!ELEMENT point (pnr*, code*,
    x+, y+, z+)>
  <!ELEMENT pnr (#PCDATA)>
  <!ELEMENT code (#PCDATA)>
  <!ELEMENT x (#PCDATA)>
  <!ELEMENT y (#PCDATA)>
  <!ELEMENT z (#PCDATA)>
  <!ELEMENT cameraData (camera+)>
  <!ELEMENT camera (type+, ck+,
    xh+, yh+, a1+, a2+, formX+,
    formY+)>
  <!ELEMENT type (#PCDATA)>
  <!ELEMENT ck (#PCDATA)>
  <!ELEMENT xh (#PCDATA)>
  <!ELEMENT yh (#PCDATA)>
  <!ELEMENT a1 (#PCDATA)>
  <!ELEMENT a2 (#PCDATA)>
  <!ELEMENT formX (#PCDATA)>
  <!ELEMENT formY (#PCDATA)>

```

Auf die Diskussion der Details muss hier verzichtet werden. Es ist noch darauf hinzuweisen, dass die DTD zukünftig durch die leistungsfähigere Strukturbeschreibungssprache XSchema abgelöst wird.

Was haben wir nun durch Speicherung der Projektdaten in einer XML-Datei und Definition eines XSL-Stylesheets erreicht? Die Daten wurden strukturiert, getrennt von der Verarbeitung gespeichert. Die XSL-Transformation, im vorliegenden Fall die Konvertierung in das HTML-Format, dient lediglich der ansprechenden Anzeige des Datenmaterials. Es ist jedoch beabsichtigt, das Projekt zur weiteren Bearbeitung in einer Client-Server-Umgebung bzw. im Web weiteren Anwendern verfügbar zu machen. Erst im Rahmen dieser Anwendung wird dann der Vorteil einer strukturierten Datenspeicherung, getrennt von der Applikation, deutlich erkennbar. Zu berücksichtigen ist auch, dass moderne objektorientierte Programmiersprachen wie Java oder PHP die Bearbeitung von XML-Dokumenten durch Bereitstellung umfangreicher Methoden unterstützen.

4 Serverseitige Anwendungen mit PHP Hypertext Preprozessor

Funktionalitäten, die ein Webbrowser nicht beinhaltet, werden clientseitig z. B. durch JavaScript zugänglich. Unter www.programmierpraktikum.de befindet sich ein Beispiel zur Berechnung räumlicher Vorwärtseinschnitte. Im Menüpunkt JavaScript ist eine HTML-Seite aufrufbar, die das Interface zur Ein- und Ausgabe der Daten und eine in JavaScript realisierte Berechnungsfunktion bereitstellt. Es besteht aber keine Möglichkeit des Dateizugriffs, weder clientseitig noch serverseitig.

Den serverseitigen Zugriff auf die XML-Datenstruktur kann man mit der Programmiersprache PHP realisieren. PHP ist als Servererweiterung zu verstehen. Die vom Client gestellte Anfrage wird beispielsweise wie folgt übermittelt:

```

http://www.imagefact.de/zitadelle-wesel/
parse_camera.php?camera=nikon_28

```

Der Server erkennt an der Erweiterungsbezeichnung `php`, dass zunächst die HTML-Seite generiert werden muss, bevor die Seite an den anfragenden Client übermittelt wird. In der Anfrage wurde gleichzeitig das Argument `nikon_28` mit übergeben. Es wird jetzt in der XML-Datenstruktur nach den Kameradaten gesucht. Zu diesem Zweck wird das XML-Dokument geparkt. Ein Parser analysiert die Syntax eines Dokuments und listet im vorliegenden Fall die Daten auf bzw. stellt deren Gültigkeit fest. Expat ist ein in PHP integrierter Parser für XML-Dokumente. Expat ist ein ereignisorientierter Parser. Ereignisse sind z. B. das Auftreten eines öffnenden Tags, das Vorkommen von zwischen den Tags eingeschlossenen Inhalten oder ein schließender Tag. Zur Verarbeitung dieser Ereignisse werden vom Programmierer die Funktionen wie folgt definiert:

```
// start_element()-wird vom XML-Parser
// bei öffnenden Tags aufgerufen
function start_element
($parser, $name) {
    global $curr_tag;
    $curr_tag = $name;
}
// end_element()-wird vom XML-Parser
// bei schließenden Tags aufgerufen
function end_element
($parser, $name)
{
    global $curr_tag;
    $curr_tag = $name;
}
// daten()-wird vom XML-Parser für Inhalte
// aufgerufen
// hier mit Bereitstellung der Kameradaten
function inhalt ($parser, $data) {
    global $curr_tag, $index, $camera;
    switch ($curr_tag) {
        case "type" :
            $index = $data;
            break;
        case "ck" :
            $camera[$index][ck] = $data;
            break;
        case "xh" :
            $camera[$index][xh] = $data;
            break;
        case "yh" :
```

```
            $camera[$index][yh] = $data;
            break;
        case "a1" :
            $camera[$index][a1] = $data;
            break;
        case "a2" :
            $camera[$index][a2] = $data;
            break;
        case "formX" :
            $camera[$index][formX] = $data;
            break;
        case "formY" :
            $camera[$index][formY] = $data;
            break;
    }
}
```

Zum Aufruf des Parsers sind folgende Anweisungen hinreichend:

```
// Instanz des XML-Parsers erzeugen
$parser = xml_parser_create();
//
// Parameter des XML-Parsers setzen
xml_parser_set_option( $parser,
XML_OPTION_CASE_FOLDING, 0 );
//
// Handler für Elemente ( öffnende /
// schließende Tags ) setzen
xml_set_element_handler( $parser,
"start_element", "end_element" );
// Handler für Inhalte setzen
xml_set_character_data_handler(
$parser, "inhalt" );
//
// Versuchen, die Datei zu öffnen
if ( ! ($fp = fopen( "zitadelle.xml",
"r" ) ) )
{
    // Fehler -> Ausführung abbrechen
    die("XML Fehler: Konnte Datei nicht
    öffnen.");
}
//
// Datei zeilenweise auslesen
while( $data = fread( $fp, 4096 ) )
{
    // aktuelle Zeile parsen
    if ( !xml_parse( $parser, $data,
feof( $fp ) ) )
    {
        // Fehlerbehandlung
        die ( "XML Error: ".
xml_error_string(
xml_get_error_code(
$parser ) ) .
"in Zeile ".
```

```

        xml_get_current_line_number(
            $parser
        );
    }
}
// Speicher wieder freigeben
xml_parser_free( $parser );

```

Die Daten werden in der Funktion `inhalt()` im mehrdimensionalen Array `camera[][]` gespeichert. An dieser Stelle ist es notwendig, auf die Handhabung mehrdimensionaler Arrays in PHP etwas näher einzugehen. Arrays werden durch Angabe von eckigen Klammern deklariert, dabei kann auf den Index verzichtet werden, da die Elemente immer an das Ende des Arrays angehängt werden. Ein Array-Element kann wieder ein Array sein, hierdurch sind mehrdimensionale Felder deklariert. Der Array-Index muss in PHP nicht durch eine Ganzzahl bezeichnet werden, dieser Schlüssel kann eine beliebige Zeichenkette sein. Wir sprechen von assoziativen Arrays, die Elemente können z.B. mit `camera[nikon_28][ck]` referenziert werden. Dort ist dann die Kamerakonstante des mit `nikon_28` bezeichneten Aufnahmesystems abgelegt. Mit diesen Informationen sollte die Datenbereitstellung innerhalb der obigen Switch-Anweisung in der Funktion `inhalt()` transparent sein.

Die beim Client im Browser angezeigte Seite enthält nur noch die HTML-Anweisungen, die serverseitig vom PHP-Prozessor erzeugt wurden. Innerhalb eines PHP-Scripts können die HTML-Anweisungen durch die Funktion `echo()` generiert werden. Abschließend noch ein Blick auf die Verarbeitung der Kameradaten, hier als einfache Anzeige im Browser:

```

// Sortieren des Arrays nach Schlüsseln
    ksort( $camera );
// ksort, krsort (absteigend);
// Feld sortieren, Zuordnung zwischen
// Elementen und Schlüsseln erhalten
    ksort( $camera );
//
// Pointer auf den Anfang des Feldes
// setzen reset( $camera );
//
// each in Verbindung mit list durch-
// lauft das Array

```

```

while (list( $key, $element ) = each(
    $camera )) {
    echo ( "$key= $element <br/>" );
    // die Elemente sind wieder Arrays
    foreach( $element as $key => $value )
        { // Elemente der Arrays ausgeben
            echo ( "$key = $value <br/>" );
        }
}

```

Anhand dieser wenigen Anweisungen wird die Mächtigkeit von PHP deutlich. Mit der `list()` Funktion wird das Feld `camera` durchlaufen, Schlüssel sind hier die Bezeichnungen der Aufnahmesysteme (*nikon*, *kodak* usw.). Die `for-each` Anweisung stellt Schlüssel `ck`, `xh`, `yh` usw. und die zugehörigen Inhalte bereit, die dann mit dem Befehl `echo()` in das HTML-Dokument geschrieben werden.

Ein weiteres Beispiel, das entsprechend dem diskutierten Quellcode leicht verifizierbar ist, stellt die Anfrage nach den Daten der äußeren Orientierung dar, vgl. auch die Angaben auf der Web-Site www.programmierpraktikum.de/pfg_publication. Das Ergebnis mit der Anbindung von JavaScript, Button mit der Beschriftung JavaScript, zeigt der



Abb. 5: Ergebnis einer Anfrage an die XML Datenbank, ausgeführt durch ein serverseitiges PHP-Script.

Browser wie in der Abb. 5 dargestellt an. Mit JavaScript kann dann clientseitig im Bild gemessen werden und nach Maßgabe der hinterlegten Modelldaten die weitere Objektbestimmung erfolgen. Die Speicherung der Daten erfolgt dann wieder serverseitig, clientseitig ausgelöst durch ein zugeordnetes Ereignis.

5 Grafik mit SVG und X3D

Die grafische Detailauswertung einer Architekturaufnahme wird üblicherweise auf der Basis der Rohdaten von Detailauswertungen mit einem CAD-System erfolgen. Ergebnisse sind die Darstellung des Bauwerks in Ansichten und Schnitten und auch als 3D-Modell für Rekonstruktionen, vgl. Abb. 1, fotorealistische Computergrafiken, Animationen und interaktive Betrachtung in VR-Umgebungen.

Mit SVG (Scalable Vector Graphics) liegt ein Web-Standard für zweidimensionale Grafiken, formuliert in XML, vor. Vektorgrafiken haben gegenüber Rastergrafiken den Vorteil, dass ohne Qualitätsverlust eine beliebige Vergrößerung möglich ist. Neben der Darstellung von geometrischen Formen können Rastergrafiken und Grafikeffekte in SVG-Dokumente eingebunden werden. Die Elemente können mit Hyperlinks ausgestattet sein. Durch Interaktion und Animation sind Präsentationsmöglichkeiten gegeben, die dem Medium Internet gerecht werden. Voraussetzung zur Betrachtung von SVG-Dokumenten ist die Einbindung eines SVG-Viewers in den Web-Browser. Derzeit ist der SVG-Viewer von Adobe populär.

Abb. 6 zeigt einen Ausschnitt aus der Übersicht der Messbildanordnung. Die Kamerasymbole enthalten in Form eines Hyperlinks die oben beschriebene Anforderung des Messbildes und der zugehörigen Daten der äußeren Orientierung. Bei Klick auf das Kamerasymbol öffnet sich ein separates Fenster, in dem dann bedarfsweise weitere photogrammetrische Auswertungen vorgenommen werden können.

Ohne zu sehr ins Detail zu gehen, soll noch kurz die Konvertierung der CAD-Daten in ein SVG Format erörtert werden.



Abb. 6: Messbildanordnung (Ausschnitt).

Quelldatei ist ein DXF-Format, das u. a. aus den Sektionen `Blocks` und `Entities` besteht. Unser Konverter berücksichtigt die, in beiden Sektionen vorkommenden Grafikelemente Linie, Ellipsenbogen (Kreis) und Polylinien. Innerhalb von Polylinien werden Ellipsenbogen im DXF-Format als sog. Bulges behandelt. Die Umsetzung dieser Elemente in SVG-Pfade erfordert einen gewissen Programmieraufwand und die Kenntnis der geometrischen Beziehungen. Eine detaillierte Darstellung findet man unter <http://www.afralisp.com/lisp/Bulges1.htm>. Die Attributzuordnung `Layer`, die im DXF-Format als Selektionskriterium und zur Bündelung von Gestaltungsattributen nutzbar ist, wird in die SVG-Datei als `Class`-Attribut übertragen. Die Ausgestaltung einer SVG-Grafik erfolgt durch die bereits erörterten Stylesheets und erlaubt daher eine sehr flexible Handhabung. Blöcke aus der DXF-Datei werden auch im SVG-Dokument, hier als Gruppen, beibehalten. Auf diese Gruppen können Transformation, Skalierung und Rotation angewandt werden, vgl. Kamerasymbole in Abb. 6. Somit wird auch die Entität `Insert` einer DXF-Datei bei der Konvertierung berücksichtigt. Geschlossenen Linienzüge, die in SVG mit `Path` bezeichnet werden, sind auch mit Flächenfüllung durch Muster (`Pattern`) oder Farbe zu gestalten. Abb. 7 zeigt die exemplarische Auswertung der hofseitigen Fassadenansicht der Zitadelle in Wesel. Auf der Webseite www.imagefact.de/zitadelle-wesel kann



Abb. 7: Photogrammetrische Auswertung der hofseitigen Fassade im SVG-Format.

man in die Grafik ein- und auszoomen und weitere Viewer-Funktionen durch Aufruf des Kontext-Menüs (rechte Maustaste) nutzen. Unter anderem steht hier auch eine Funktionalität zur komprimierten Speicherung komplexer Dateien zur Verfügung.

Für 3D-Grafiken im Web galt bisher das Format VRML Virtual Reality Modeling Language, mit der Dateierweiterungsbezeichnung `.wrl` oder `.wrlz` für komprimierte Dateien. Dieses Format wird zukünftig ebenfalls durch eine XML-Formulierung ersetzt. Der Standard ist dann die DTD für extensible 3D (X3D). Unter Windows ist mit Octaga ein 3D Viewer, der stand-alone oder als Plug-in die Formate VRML und X3D anzeigen kann, verfügbar. Ein auf Java basierendes Konvertierungswerkzeug `Vrml97ToX3d-Nist` ist von der Web-Site des National Institute of Standards and Technology zu beziehen. Zur Vertiefung weiterer Details und zur interaktiven Navigation durch die Zitadelle Wesel sollte der Leser die, in den Referenzen angegebenen Webseiten besuchen.

6 Zusammenfassung

XML und zugehörige Technologien sind die Grundlage von Web-Applikationen der zweiten Generation. Während bisherige Anwendungen auf der Basis von HTML die Gestaltung von Publikationen zur Zielstellung hatten, steht künftig die strukturierte Speicherung von Informationen, Interak-

tion und verteilte Weiterverarbeitung der Daten im Fokus. Eine exemplarische Anwendung der Architekturphotogrammetrie wurde zur Skizzierung der Möglichkeiten und Einführung in die Sprachen XML, XSL, PHP, SVG und X3D herangezogen. Aus der Sicht des Endanwenders kann besonders die Formulierung von Grafikdaten in SVG eine Innovation darstellen. Einerseits bietet die „Zeichnungsausgestaltung“ mit Interaktion, Animation und Hyperlinks das adäquate Spektrum für multimediale Umgebungen, darüber hinaus ist die Bedeutung von Scalable Vector Graphics auch für Printmedien nicht unerheblich.

Softwarehersteller werden ihre Datenstrukturen in XML formulieren, nicht zuletzt auch wegen der mächtigen Unterstützung durch objektorientierte Programmiersprachen wie Java oder PHP. Möglicherweise wird dann in den Normungsausschüssen auch über die Formulierung von DTDs für photogrammetrische Applikationen entschieden.

Diese Arbeit entstand im Rahmen eines Forschungsaufenthaltes von Dr. NIKOLAY M. DEMENTIEV, Universität Wologda, Russland am Fachbereich Architektur und Bauingenieurwesen der Fachhochschule Bielefeld in Zusammenarbeit mit dem Preußen-Museum NRW. Gefördert wurde das Vorhaben durch ein Michail-Lomonosov-Stipendium des DAAD.

Literatur

- POMASKA, G., 2005: Grundkurs Web-Programmierung. – Vieweg-Verlag, Wiesbaden.
- POMASKA, G., 2003: Introduction of SVG as a data interchange format for architectural documentations. – CIPA International Symposium, Antalya, Turkey.
- POMASKA, G., 2003: Implementation of Web 3D tools for creating interactive walk-through environments from building documentations. – ISPRS WG V/4 and IC WG V/III International Workshop on Vision Techniques for Digital Architectural and Archaeological Archives, Ancona, Italy.
- VELTZKE, V., 2001: Preußische Festung Wesel. – Reihe „Der historische Ort“, Nr. 87, Kai Homilius Verlag, Berlin.
- <http://www.imagefact.de/zitadelle-wesel>
Website zum Projekt Bauaufnahme der Zitadelle in Wesel.
- http://www.programmierpraktikum.de/pfg_publication
Die Website zur Publikation mit Programmierbeispielen.
- http://ovrt.nist.gov/v2_x3d.html
National Institute of Standards and Technology, Konvertierung zwischen VRML 97 und X3D.
- <http://www.octaga.com>
Octaga VRML/X3D Viewer
- <http://www.sigma-foto.de/>
Sigma Kameras und Objektive
- <http://www.afralisp.com/lisp/Bulges1.htm>
Beschreibung von Polyline Bulges.
- <http://www.adobe.com/svg>
Download SVG-Viewer

Anschriften der Autoren:

Prof. Dr.-Ing. GÜNTER POMASKA
FH Bielefeld,
Labor für visuelle und virtuelle Realität
Artilleriestr. 9
D-32427 Minden
e-mail: gp@imagefact.de
<http://www.divide-by-zero.com>

Dr. NIKOLAY M. DEMENTIEV
Staatliche Technische Universität Wologda, Lehrstuhl für Industrie- und Zivilbau
Lenin Str. 15, 160035 Wologda, Russland
e-mail: ndementiev@mh.vstu.edu.ru

Manuskript eingereicht: Februar 2005
Angenommen: April 2005