3D – Filmauswertung bei mitgeschwenkter und gezoomter Kamera

Martin Mössner
Institut für Mathematik und Geometrie
Universität Innsbruck

Institutsbericht August 1992

Diese Arbeit wurde vom Fonds zur Förderung der wissenschaftlichen Forschung unterstützt.

Abstract

The Direct Linear Transformation and an extension are used to get the relationship between given planar film-data and the object coordinates of a moving body. The object moves through a field of calibration passpoints and is filmed by two synchronized cameras. In each frame at least 6 passpoints are necessary. Calibration is done independently for every pair of frames. Therefore, passpoints can change from frame to frame and from camera to camera. The cameras can be panned and tilted and the lenses can be zoomed. The overdetermined system of equations is solved by least squares. Data errors can be found by large residuals. A computer program has been developed in Fortran 90 and is available in Fortran 77 and Ansi C too.

Kontaktadresse:
Mag. Martin Mössner
Institut für Mathematik und Geometrie
Technikerstrasse 13
A – 6020 Innsbruck, Austria
email: martin.moessner@uibk.ac.at
Tel.: 0512/218-4167
Fax: 0512/218-4036

Inhaltsangabe

1. Optische Grundlagen

Die Direct Linear Transformation (DLT) wurde von Abdel-Aziz und Karara (1971) vorgestellt. Ihre allgemeine Form ist eine perspektivische Abbildung von $\mathbf{R}^3 \backslash \{(X,Y,Z)^t \mid b_9X + b_{10}Y + b_{11}Z + 1 = 0\}$ in $\mathbf{R}^2$:

$$\begin{aligned} x &= \frac{b_1X + b_2Y + b_3Z + b_4}{b_9X + b_{10}Y + b_{11}Z + 1} \\ y &= \frac{b_5X + b_6Y + b_7Z + b_8}{b_9X + b_{10}Y + b_{11}Z + 1}. \end{aligned} \tag{1}$$

x, y sind die Bildkoordinaten und X, Y, Z die Objektkoordinaten. $(b_1, \ldots, b_{11})$ bezeichnen wir als Kalibrierungsvektor oder DLT-Parameter.

Sind die 11 DLT-Parameter $b_1, \ldots, b_{11}$ dieser Abbildung für mindestens zwei synchrone Bilder bekannt, so können die Objektkoordinaten aus den entsprechenden Bildkoordinaten bestimmt werden.

Die Darstellung (1) der DLT läßt sich leicht aus der Theorie der geometrischen Optik (siehe z.B. Hecht, Zajac 1974) gewinnen.

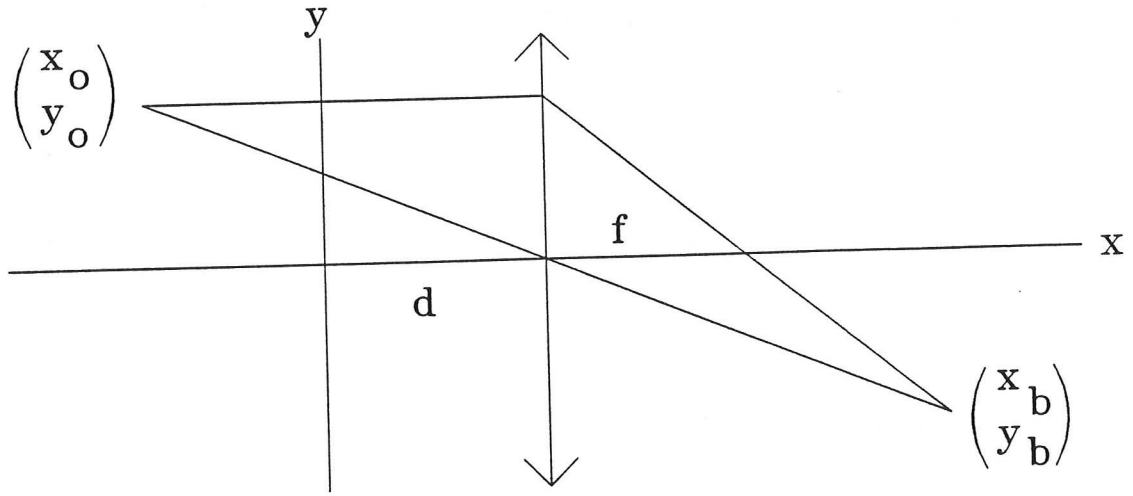

Abb. 1: Strahlengang bei dünnen Linsen

Mit Hilfe des Strahlensatzes

$$(d - x_o) : y_o \quad = \quad (x_b - d) : (-y_b) \tag{2}$$

und des Brechungsgesetzes

$$\frac{1}{d - x_o} + \frac{1}{x_b - d} = \frac{1}{f} \tag{3}$$

erhält man aus den Objektkoordinaten x_o, y_o die Bildkoordinaten x_b, y_b :

$$\begin{aligned} x_b &= \frac{fx_o + d(x_o - d)}{f + x_o - d} \\ y_b &= \frac{fy_o}{f + x_o - d}. \end{aligned} \tag{4}$$

Die Nenner der linear gebrochenen Abbildungen (4) sind gleich. Die optische Abbildung durch eine dünne Linse ist daher eine Perspektivität:

$$\begin{pmatrix} x_b \\ y_b \end{pmatrix} = f \begin{pmatrix} x_o \\ y_o \end{pmatrix}, \qquad f \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{c_7 x + c_8 y + c_9} \begin{pmatrix} c_1 x + c_2 y + c_3 \\ c_4 x + c_5 y + c_6 \end{pmatrix} \tag{5}$$

Die Parameter c_i ergeben sich aus (4).

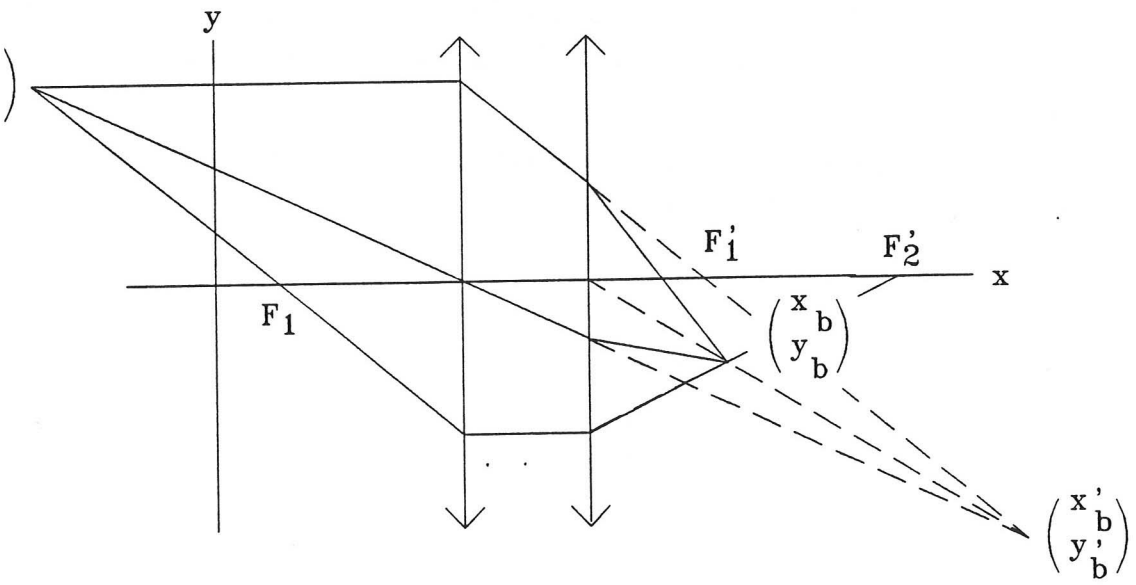

Abb. 2: dicke Linsen

Eine dicke Linse besteht aus der Aneinanderreihung zweier dünner Linsen, wobei zu beachten ist, daß die zwei Brennweiten entgegengesetztes Vorzeichen haben und die optisch aktiven Ebenen gegeneinander verschoben sind.

Kameras (Film- oder Videokameras) bestehen aus Linsensystemen, d.h. aus der Aneinanderreihung mehrerer dicker und dünner Linsen. Als Gesamtabbildung erhält man die Hintereinanderausführung endlich vieler Abbildungen der Gestalt von Gleichung (5). Durch kurze Rechnung überzeugt man sich leicht, daß die Hintereinanderausführung zweier Abbildungen, wie in (5) gegeben, wieder eine Abbildung der Gestalt von (5) ergibt, wobei allerdings andere Koeffizienten auftreten werden. Verdreht und

verschiebt man schließlich noch Objekt- und Bildkoordinaten, so erhält man die in (1) gegebene Darstellung der DLT.

Bei dieser Vorgangsweise ist es möglich, daß die Filmdaten zwecks Auswertung ein weiteres mal projiziert werden. Eine solche Projektion wird ebenfalls durch eine Abbildung der Gestalt (5) beschrieben und kann so problemlos mitbehandelt werden. Es ist allerdings zu beachten, daß eine direkte physikalische Interpretation der DLT-Parameter b_i, so wie sie von Abdel-Aziz (1971), Marzan, Karara (1975) oder anderen gegeben werden, nicht mehr möglich ist.

Neben der eigentlichen DLT-Transformation (1) wurde eine Erweiterung, die von Karara und Marzan (1975) vorgeschlagen wurde, implementiert. Um Linsenfehler, wie Aberration, Distorsion, Koma und ähnliches, auszugleichen, werden Korrekturen Δx und Δy eingeführt:

$$\begin{aligned} x &= \frac{b_1 X + b_2 Y + b_3 Z + b_4}{b_9 X + b_{10} Y + b_{11} Z + 1} + \Delta x \\ y &= \frac{b_5 X + b_6 Y + b_7 Z + b_8}{b_9 X + b_{10} Y + b_{11} Z + 1} + \Delta y. \end{aligned} \tag{6}$$

Damit können Effekte berücksichtigt werden, die von der Wellenlänge unabhängig sind. Für die Abweichungen Δx und Δy werden 3 symmetrische und zwei asymmetrische Terme hinzugefügt:

$$\begin{aligned} \Delta x &= u(b_{12} r + b_{13} r^2 + b_{14} r^3) + b_{15}(r + 2u^2) + 2b_{16} uv \\ \Delta y &= v(b_{12} r + b_{13} r^2 + b_{14} r^3) + 2b_{15} uv + b_{16}(r + 2v^2). \end{aligned} \tag{7}$$

Die Parameter $b_{12}, \ldots, b_{16}$ können folgenderweise interpretiert werden:

b_{12}, b_{13}, b_{14}	symmetrische Linsendistorsion
b_{15}, b_{16}	asymmetrische Linsendistorsion.

Die Größen r, u und v sind Relativkoordinaten zum optischen Zentrum x_0, y_0. Es gilt nach Karara und Marzan (1975):

$$\begin{aligned} r &= u^2 + v^2 \qquad u = x - x_0 \qquad v = y - y_0 \\ x_0 &= \frac{b_1 b_9 + b_2 b_{10} + b_3 b_{11} + b_4}{b_9^2 + b_{10}^2 + b_{11}^2 + 1} \\ y_0 &= \frac{b_5 b_9 + b_6 b_{10} + b_7 b_{11} + b_8}{b_9^2 + b_{10}^2 + b_{11}^2 + 1}. \end{aligned} \tag{8}$$

2. DLT-Parameter

Zuerst wird für jede Kamera und jedes Bild der Kalibrierungsvektor bestimmt. Dazu multipliziert man (1) mit dem Nenner:

$$\begin{aligned} b_1X + b_2Y + b_3Z + b_4 - b_9xX - b_{10}xY - b_{11}xZ - x = 0 \\ b_5X + b_6Y + b_7Z + b_8 - b_9yX - b_{10}yY - b_{11}yZ - y = 0. \end{aligned} \tag{9}$$

Wir nehmen an, daß für das jeweilige Bild die Bildkoordinaten x_i, y_i und die Ortskoordinaten X_i, Y_i, Z_i, $1 \leq i \leq n$ von n Paßpunkten bekannt sind. Einsetzen in (9) liefert:

$$Ab = r \tag{10}$$

$$A = \begin{pmatrix} X_i, & Y_i, & Z_i, & 1, & 0, & 0, & 0, & 0, & -x_iX_i, & -x_iY_i, & -x_iZ_i \\ 0, & 0, & 0, & 0, & X_i, & Y_i, & Z_i, & 1, & -y_iX_i, & -y_iY_i, & -y_iZ_i \end{pmatrix}_{i=1,\dots,n}$$

$$b = (b_1, \dots, b_{11})^t \qquad r = (x_i, y_i)^t_{i=1,\dots,n}.$$

Um dieses Gleichungssystem für die 11 Unbekannten $b_1, \dots, b_{11}$ zu lösen, benötigt man mindestens 6 Paßpunkte, d.h. 12 Gleichungen. Da wir keine Information durch willkürliches Weglassen einer Gleichung verlieren wollen, bestimmen wir die Lösung, die die kleinste Fehlerquadratsumme aufweist. Falls A vollen Rang hat, d.h. Rang A = 11, ist die Lösung b des linearen Ausgleichsproblems

$$\|Ab - r\| = \min\ ! \tag{11}$$

eindeutig bestimmt. Man bezeichnet b auch als die Lösung des (überbestimmten) Gleichungssystems (10). Falls vier Paßpunkte komplanar sind, so sind zwei Zeilen von A linear abhängig und können daher weggelassen werden. Unter den n Paßpunkten müssen sich mindestens 6 Paßpunkte befinden, von denen jeweils 4 nicht komplanar sind.

Die Ausgleichslösung erhält man mit Hilfe des QR- oder des QL-Algorithmus. Codes dafür sind in den Fortranbibliotheken Lapack (1992) oder Linpack (1979) zu finden. Wir verwenden die Programme decqr und solqr von Hairer (1991). Durch Berechnung der Ausgleichslösung kann man einerseits Meßfehler in den Bild- und Ortskoordinaten der Paßpunkte glätten und andererseits Datenfehler in den Paßpunkten erkennen. Aus diesem Grund empfiehlt es sich, mehr als 6 Paßpunkte zu verwenden.

Die Parameter der erweiterten DLT-Transformation kann man nicht mehr direkt durch Lösen eines linearen Gleichungssystems bestimmen, da diese durch Einführung der optischen Achse nichtlinear auftreten. Weil Δx und Δy optische Korrekturen darstellen, ist folgender Ansatz sinnvoll:

1) bestimme $b_1, \ldots, b_{11}$ aus (10)
2) berechne die optische Achse nach (8)
3) berechne $b_1, \ldots, b_{16}$ durch Lösen des linearen Gleichungssystems (13)
4) gehe nach 2) solange sich b ändert.

Nehme als Startwert für $b_1^{(k)}, \ldots, b_{11}^{(k)}$ $k = 0$, die Lösung von (10) bzw. (11). Berechne die Näherung für die optische Achse x_0, y_0 :

$$\begin{aligned} l^{(k)} &= \frac{1}{(b_9^{(k)})^2 + (b_{10}^{(k)})^2 + (b_{11}^{(k)})^2 + 1} \\ x_0^{(k)} &= (b_1^{(k)} b_9^{(k)} + b_2^{(k)} b_{10}^{(k)} + b_3^{(k)} b_{11}^{(k)} + b_4^{(k)}) l^{(k)} \\ y_0^{(k)} &= (b_5^{(k)} b_9^{(k)} + b_6^{(k)} b_{10}^{(k)} + b_7^{(k)} b_{11}^{(k)} + b_8^{(k)}) l^{(k)} \end{aligned} \tag{12a}$$

und für die Paßpunkte $i = 1, \ldots, n$ die Relativkoordinaten u_i, v_i, r_i :

$$\begin{aligned} a_i^{(k)} &= \frac{1}{b_9^{(k)} X_i + b_{10}^{(k)} Y_i + b_{11}^{(k)} Z_i + 1} \\ u_i^{(k)} &= x_i - x_0^{(k)} \\ v_i^{(k)} &= y_i - y_0^{(k)} \\ r_i^{(k)} &= (u_i^{(k)})^2 + (v_i^{(k)})^2. \end{aligned} \tag{12b}$$

Löse damit das (überbestimmte) lineare Gleichungssystem

$$A^{(k)} b^{(k+1)} = r^{(k)} \tag{13}$$

$$(A^{(k)})^t = \begin{pmatrix} X_i a_i^{(k)}, & 0 \\ Y_i a_i^{(k)}, & 0 \\ Z_i a_i^{(k)}, & 0 \\ a_i^{(k)}, & 0 \\ 0, & X_i a_i^{(k)} \\ 0, & Y_i a_i^{(k)} \\ 0, & Z_i a_i^{(k)} \\ 0, & a_i^{(k)} \\ -x_i X_i a_i^{(k)}, & -y_i X_i a_i^{(k)} \\ -x_i Y_i a_i^{(k)}, & -y_i Y_i a_i^{(k)} \\ -x_i Z_i a_i^{(k)}, & -y_i Z_i a_i^{(k)} \\ u_i^{(k)} r_i^{(k)}, & v_i^{(k)} r_i^{(k)} \\ u_i^{(k)} (r_i^{(k)})^2, & v_i^{(k)} (r_i^{(k)})^2 \\ u_i^{(k)} (r_i^{(k)})^3, & v_i^{(k)} (r_i^{(k)})^3 \\ r_i^{(k)} + 2(u_i^{(k)})^2, & 2u_i^{(k)} v_i^{(k)} \\ 2u_i^{(k)} v_i^{(k)}, & r_i^{(k)} + 2(v_i^{(k)})^2 \end{pmatrix}_{i=1,\dots,n}$$

$$b^{(k+1)} = (b_1^{(k+1)}, \dots, b_{16}^{(k+1)})^t \qquad r^{(k)} = (x_i a_i^{(k)}, y_i a_i^{(k)})_{i=1,\dots,n}^t.$$

Ersetze k durch $k+1$ und wiederhole diesen Schritt solange, bis eine vorgegebene Toleranz Tol unterschritten wird

$$\|b^{(k)} - b^{(k-1)}\| < Tol \tag{14}$$

oder bis die maximale Anzahl von Iterationen erreicht ist. Im Falle des Versagens der Iteration nehme die 11 Parameter-Auswertung und setze $b_{12} = \dots = b_{16} = 0$.

Im Gegensatz zur 11-Parameter Auswertung hat man hier 16 Unbekannte, sodaß mindestens 8 Paßpunkte notwendig sind. Verwendet man mehr als 8 Paßpunkte, so kann man, wie im einfachen Fall, durch Bestimmung der Ausgleichslösung Meßfehler glätten und Datenfehler in den Paßpunkten erkennen.

3. Bildauswertung

Wir nehmen an, daß synchrone Bilder für m Kameras ($m \geq 2$) vorliegen. Wir bezeichnen mit x_j, y_j die Bildkoordinaten des gesuchten Punktes mit den Objektkoordinaten $p = (X, Y, Z)$. Der Index j bezieht sich auf die j-te Kamera ($j = 1, \ldots, m$).

Für die Bildauswertung zieht man wieder die Grundgleichungen (1) bzw. (6) heran. Im Fall der 16-Parameter Auswertung berechnet man in einem ersten Schritt für die j-te Kamera ($j = 1, \ldots, m$) die Korrekturen Δx^j und Δy^j nach (7).

$$
\begin{aligned}
x_0^j &= \frac{b_1^j b_9^j + b_2^j b_{10}^j + b_3^j b_{11}^j + b_4^j}{(b_9^j)^2 + (b_{10}^j)^2 + (b_{11}^j)^2 + 1} \\
y_0^j &= \frac{b_5^j b_9^j + b_6^j b_{10}^j + b_7^j b_{11}^j + b_8^j}{(b_9^j)^2 + (b_{10}^j)^2 + (b_{11}^j)^2 + 1} \\
u^j &= x_j - x_0^j \\
v^j &= y_j - y_0^j \\
r^j &= (u^j)^2 + (v^j)^2 \\
\Delta x^j &= u^j(b_{12}^j r^j + b_{13}^j (r^j)^2 + b_{14}^j (r^j)^3) + b_{15}^j (r^j + 2(u^j)^2) + 2b_{16}^j u^j v^j \\
\Delta y^j &= v^j(b_{12}^j r^j + b_{13}^j (r^j)^2 + b_{14}^j (r^j)^3) + 2b_{15}^j u^j v^j + b_{16}^j (r^j + 2(v^j)^2).
\end{aligned}
\tag{15}
$$

Man korrigiert die Bilddaten, indem man x_j durch $x_j - \Delta x^j$ und y_j durch $y_j - \Delta y^j$ ersetzt. Es bleibt nur mehr Gleichung (1) zu invertieren. Aus (9) folgt

$$
Ap = c \tag{16}
$$

$$
A = \begin{pmatrix} b_1^j - x_j b_9^j, & b_2^j - x_j b_{10}^j, & b_3^j - x_j b_{11}^j \\ b_5^j - y_j b_9^j, & b_6^j - y_j b_{10}^j, & b_7^j - y_j b_{11}^j \end{pmatrix}_{j=1,\ldots,m}
$$

$$
p = (X, Y, Z)^t \qquad c = (x_j - b_4^j, y_j - b_8^j)_{j=1,\ldots,m}^t.
$$

Hier sind

b^j	Kalibrierungsvektor der Kamera j
x_j, y_j	(korrigierte) Filmkoordinaten des auszuwertenden Körperpunktes X, Y, Z von Kamera j

Für $m \geq 2$ Kameras hat man wieder ein überbestimmtes lineares Gleichungssystem für die Unbekannten X, Y, Z.

4. Skalierung

Zur Bildauswertung müssen Gleichungssysteme vom Typ (10), (13) und (16) gelöst werden. Wir nehmen an, daß diese Gleichungssysteme vollen Rang besitzen. Im folgenden untersuchen wir, wie sich Meßfehler auf den mit Gleichung (10) bestimmten Kalibrierungsvektor auswirken.

Die Genauigkeit der numerischen Lösung eines solchen (überbestimmten) linearen Gleichungssystems kann mit Hilfe der Kondition abgeschätzt werden (Golub, van Loan 1990). Hierzu vergleichen wir die exakte Lösung x eines ungestörten Gleichungssystems

$$Ax = b \tag{17}$$

mit der exakten Lösung $x + \Delta x$ eines gestörten Gleichungssystems

$$(A + \Delta A)(x + \Delta x) = b + \Delta b. \tag{18}$$

Die Störungen können Fehler in der Zahlendarstellung sein, oder, was für unsere Anwendung wichtiger ist, von Meßfehlern hervorgerufen werden. Bei uns treten relative Meßfehler ϵ in der Größenordnung von 10^{-3} bis 10^{-5} auf. Durch Einführung der Kondition

$$\kappa(A) = \|A\|\|(A^tA)^{-1}A^t\|, \tag{19}$$

und des Residuums $\rho = \|Ax - b\|$ kann man eine Abschätzung für den relativen Fehler der numerischen Lösung in der euklidischen Norm angeben (Golub, van Loan 1990):

$$\begin{aligned} &\frac{||\Delta x||}{||x||} \leq \left(\frac{2\kappa(A)}{\cos\vartheta} + \tan\vartheta\,\kappa(A)^2\right)\epsilon + O(\epsilon^2) \\ \text{falls} \qquad &\epsilon := \max\left(\frac{||\Delta A||}{||A||}, \frac{||\Delta b||}{||b||}\right) < \frac{1}{\kappa(A)} \qquad \sin\vartheta = \frac{\rho}{||b||}. \end{aligned} \tag{20}$$

Realistische Werte sind $|\sin\vartheta| < 0.01$ und $\epsilon = 10^{-4}$, sodaß man (20) in der Form

$$\frac{||\Delta x||}{||x||} \leq 2 * 10^{-4}\kappa(A) + 10^{-6}\kappa(A)^2 \tag{21}$$

verwenden kann.

Numerische Tests ergaben, daß ohne Skalierung die Kondition in praktischen Fällen 10^9 erreichen kann, sodaß eine geeignete Skalierung unerläßlich ist. Als geeignete Skalierung stellte sich folgende Vorgangsweise heraus: Verwende für die Ortskoordinaten ein Koordinatensystem, dessen Ursprung im Schwerpunkt der verwendeten Paßpunkte

liegt. Skaliere die Bildkoordinaten so, daß sie zwischen -1 und 1 sind. Liegen die Bildkoordinaten zwischen b_{min} und b_{max}, so hat man die Skalierung

$$p_{neu} = 2\,\frac{p_{alt} - b_{min}}{b_{max} - b_{min}} - 1. \tag{22}$$

Nach dieser Transformation war die Kondition von A in den untersuchten Beispielen kleiner als 200.

Für eine direkte Lösung der Normalgleichungen $A^tAx = A^tb$ erscheint uns dieser Wert zu groß. Falls A^tA exakt, d.h. mit doppelter Genauigkeit, berechnet wird, sind die relativen Fehler von A^tA und A^tb ebenfalls in der Größenordnung von ϵ. Der relative Fehler bei der Lösung der Normalgleichungen kann wie folgt abgeschätzt werden (Schwarz 1986):

$$\frac{||\Delta x||}{||x||} \leq \frac{2\kappa(A)^2}{1 - \kappa(A)^2\epsilon}\,\epsilon \qquad \text{falls} \qquad \kappa(A)^2\epsilon < 1. \tag{23}$$

Bei Meßfehlern von $\epsilon = 10^{-4}$ ist $\kappa(A)^2\epsilon \geq 1$ nicht auszuschließen.

5. Automatische Fehlererkennung

Für die automatische Fehlererkennung hat man drei Ansatzpunkte:

(1) Nach Bestimmung des Kalibrierungsvektors $(b_1, \ldots, b_{11})$ bzw. $(b_1, \ldots, b_{16})$: Berechne mit Hilfe der b_i und der Ortskoordinaten X, Y, Z der Paßpunkte die dazugehörigen Bildkoordinaten nach (1) und vergleiche diese mit den gemessenen Bildkoordinaten der Paßpunkte. Bei großen Unterschieden ist ein Datenfehler der Bildkoordinaten der Paßpunkte gegeben.

(2) Bei Filmauswertung:
Unerwartet große Residuen bei der Lösung des Gleichungssystems (16) deuten auf Datenfehler in den Bildkoordinaten hin.

(3) Nach Auswertung:
Gesamtauswertung von geodätisch vermessenen Objekten, wie Paßpunkten oder anderen Objekten, die auf beiden Filmen zu sehen sind, können dazu verwendet werden um die Größenordnung des Auswertungsfehlers zu bestimmen.

6. Programmbeschreibung von film3d

Zur 3D-Auswertung mit 2 Kameras wurde das Unterprogramm *film3d* entwikkelt. Es gibt Fortran- und C-Versionen. Als Eingabedateien werden die Bilddaten der zwei Kameras in *film3d.b*1 und *film3d.b*2, die geodätisch vermessenen Daten der Paßpunkte in *film3d.pp* und die Synchronisationsdaten in *film3d.t*1 und *film3d.t*2 benötigt. Bei der Filmanalyse werden die auftretenden Koordinaten skaliert, sodaß man die transformierten Koordinaten der Paßpunkte benötigt. Sie werden in die Datei *film3d.ppn* geschrieben. Die Ergebnisse der Filmanalyse stehen in *film3d.xyz*. Der genaue Aufbau der Ein- und Ausgabedateien ist im Kommentarteil des Programmes beschrieben.

Bei Aufruf von *film3d* müssen die Parameter *nmess* (Anzahl der insgesamt vermessenen Paßpunkte), *nbild* (maximale Anzahl der auszuwertenden Bilder), *nkalib* (maximale Anzahl von Paßpunkten, die pro Bildauswertung verwendet werden), *nkoerp* (Anzahl der Körperpunkte, die pro Bildpaar ausgewertet werden) übergeben werden. Zusätzlich müssen die Felder $work(lwork)$ und $iwork(liwork)$ mit den dazugehörigen Dimensionen *lwork* und *liwork* zur Verfügung gestellt werden. Die ersten Komponenten der Arrays *work* und *iwork* können dazu verwendet werden, um die Abarbeitung der Filmdaten zu steuern. Es sind dies $iout = iwork(1)$ (steuert zusätzlichen Output), $par = iwork(2)$ (Anzahl der Parameter für die DLT, das ist 11 oder 16), $scale = iwork(3)$ (Skalierungsart), $itmax = iwork(4)$ (maximale Anzahl der Iterationen bei der 16-Parameter Auswertung), sowie $bmin = work(1)$, $bmax = work(2)$ (Wertebereich der Bilddaten für geeignete Skalierung). Werden keine Werte für diese Größen angegeben, so werden Defaultwerte verwendet. Ist $iout > 0$, so werden die Dateien *film3d.on* mit $n = 1, \ldots, \min(6, itmax)$ geschrieben. Sie enthalten Test- und Zwischenergebnisse der Filmanalyse. Die genaue Beschreibung dieser Variablen befindet sich ebenfalls im Kommentarteil des Unterprogrammes *film3d*.

Um Fehler in den Daten zu erkennen, werden bei der Filmauswertung Paßpunkte, die von beiden Kameras gesehen werden, ausgewertet. Die Anzahl der Paßpunkte pro Bildpaar, die von beiden Kameras sichtbar sind, aufsummiert über alle Bildpaare steht in $ires = iwork(5)$. $res1 = work(3)$ ist die 1-Norm dividiert durch $ires$, $res2 = work(4)$ die 2-Norm dividiert durch $\sqrt{ires}$ und $resm = work(5)$ die Maximumsnorm der dabei auftretenden Residuen. Ist $resm$ deutlich größer als $res1$, so ist ein Datenfehler gegeben.

Hauptprogramm zum Aufruf von film3d

```
program test
integer, parameter :: &
   nmess = 46, nbild = 113, nkalib = 10, nkoerp = 2, par = 11, &
   liwork = 2 * nbild + 2 * nkalib + par + 3, &
   lwork = 3 * nmess + 2 * nbild + 14 * nkalib + &
      4 * par * nkalib + 4 * par + 7 * nkoerp + 2
integer, dimension(liwork) :: iwork
double precision, dimension(lwork) :: work
!
iwork(1) = 54     ! schreibe die Dateien
                  ! film3d.o1 film3d.o2 film3d.o3 film3d.o4
iwork(2) = 11     ! 11-Parameter Auswertung (default)
iwork(3) = 0      ! automatische Skalierung
iwork(4) = 0      ! Standardwert 10, hier ueberfluessig
work(1) = 0; work(2) = 600 ! Wertebereich fuer Bilddaten aus PEAK
!
call film3d(nmess,nkalib,nbild,nkoerp,lwork,work,liwork,iwork)
!
write(*,*)
write(*,*) ' ires ', iwork(5)
write(*,*) ' res1 ', work(3)
write(*,*) ' res2 ', work(4)
write(*,*) ' resm ', work(5)
end
```

Eingabefile film3d.pp

```
!  passpunkte der schanze in seefeld (siehe abb.3 im anhang)
!  es werden 9 zeilen kommentar und dann nmess passpunkte aus
!  der datei film3d.pp eingelesen
!  leseformat:       do n = 1, nmess
!                       read(nfahrt,*) i, j, y, x, z
!                    end do
!  i        nummer der passpunkte von den vermessern vergeben
!  j        numerierung fuer die bildauswertung
!  y, x, z  geodaetisch vermessene koordinaten fuer die passpunkte
 1      44        984.206    1003.699     999.905
 2      43        976.671    1006.335     999.776
 3      21        969.386    1008.775     999.800
 4      20        963.589    1015.189    1000.048
```

5	19	955.724	1016.938	1000.585
6	18	948.100	1018.569	1002.146
7	17	940.732	1020.344	1004.217
8	16	933.346	1021.969	1006.761
9	15	926.061	1023.488	1010.314
10	14	919.210	1025.044	1014.481
11	13	912.735	1026.581	1018.973
12	12	908.778	1027.517	1021.813
13	11	904.856	1028.383	1024.760
14	10	900.930	1029.323	1027.616
15	9	897.070	1030.324	1030.628
16	8	893.158	1031.191	1033.652
17	7	889.219	1032.158	1036.492
18	6	885.125	1033.140	1039.340
19	5	881.008	1034.030	1042.047
20	4	877.104	1034.909	1044.442
21	3	872.990	1035.881	1046.849
22	2	868.762	1036.845	1049.050
23	1	864.314	1037.784	1051.240
24	22	861.110	1028.505	1051.282
25	23	865.353	1026.891	1049.116
26	24	869.240	1025.151	1046.952
27	25	873.113	1023.602	1044.599
28	26	877.028	1021.886	1042.137
29	27	880.961	1020.313	1039.473
30	28	884.807	1018.787	1036.709
31	29	888.698	1017.180	1033.891
32	30	892.258	1015.678	1030.873
33	31	896.096	1014.273	1027.862
34	32	899.745	1012.751	1024.886
35	33	903.586	1011.188	1021.872
36	34	907.365	1009.735	1019.040
37	35	913.454	1007.363	1014.715
38	36	920.059	1004.813	1010.606
39	37	926.899	1002.319	1001.398
40	38	934.030	999.852	1004.278
41	39	941.230	997.179	1002.068
42	40	948.531	994.305	1000.608
43	41	955.975	991.521	999.941
44	42	963.401	988.535	999.615
45	45	972.405	991.671	999.831
46	46	980.003	989.123	999.893

Eingabefile film3d.b1 film3d.b2 wird analog aufgebaut

```
!
!  10 zeilen kommentar
!
!  testdaten: film3d.b1
!
!  bilddaten fuer kamera 1 mit 50 Hz
!
!  113 bilder von nummer 5 bis 117
!
!
5                  !   nummer des ersten bildes
117                !   nummer des letzten bildes
  5                !   bildnummer: bild 5
0                  !   trennzeichen '0'
 1
 2                 !   indices der in diesem bild verwendeten
 3                 !   passpunkte (siehe 2. spalte in film3d.pp)
 4
 5
23
22
24
0                  !   '0' beendet indices der passpunkte
398.50             !   es folgen die bildkoordinaten der
228.53             !   passpunkte:
423.50             !   x-koordinate, y-koordinate von passpunkt 1
194.91             !   x-koordinate, y-koordinate von passpunkt 2
447.50             !   ...
158.74             !   x-koordinate, y-koordinate von passpunkt 24
471.00
111.01
497.00
 59.66
 79.50
207.20             !   in bild 5 werden 8 passpunkte verwendet
 87.00             !   es werden 16 bildkoordinaten eingelesen
237.21             !   und dann ueberprueft, ob der naechste
 66.00             !   datensatz eine null ist.
170.32
0                  !   '0' beendet bildkoordinaten der passpunkte
231.00             !   bildkoordinaten der koerperpunkte
233.96             !   (nkoerp = 2 in test)
 90.33             !   x-koordinate, y-koordinate von koerperpunkt 1
```

```
220.17              !   x-koordinate, y-koordinate von koerperpunkt 2
  6                 !   bild 6    gleiches format wie bild 5
0
 1
 2
 3
 4
 5
23
24
0
399.50
229.62
424.50
195.99
448.00
159.83
471.00
112.46
495.50
 61.11
 80.00
208.65
 66.50
170.68
0
231.00
233.96
 70.45
180.78
...                 !   bild 7 bis bild 117
                    !   eventuell weitere bilder
                    !   werden ignoriert
```

Eingabefile: film3d.t1 film3d.t2 analog

```
1                   !   1 fuer synchronisierte bilder mit konstanter
                    !   zeitdifferenz dann leerzeile
0.1d0               !   startzeit
120 0.02d0          !   anzahl der bilder
                    !   zeitdifferenz zwischen zwei bildern
```

Ausgabefile: film3d.ppn

```
!
!   vermessene passpunkte
!   j, x, y, z
!   j          passpunktnummer
!   x, y, z    koordinaten der passpunkte im neuen system
!
!   bei der filmauswertung wird der schwerpunkt des passpunktfeldes
!   (-> mxyz) berechnet und und das ortskoordinatensystem so
!   verschoben, dass dieser schwerpunkt im ursprung liegt.
!
    1017.196       913.6613       1021.882     ! mxyz
      1    20.58754        -49.34733        29.35783
      2    19.64854        -44.89933        27.16783
      3    18.68454        -40.67133        24.96683
      4    17.71254        -36.55733        22.55983
      5    16.83354        -32.65333        20.16483
      usw.
```

Ausgabefile: film3d.xyz

```
!
!   bei der filmauswertung wird der schwerpunkt des passpunktfeldes
!   (-> mxyz) berechnet und und das ortskoordinatensystem so
!   verschoben, dass dieser schwerpunkt im ursprung liegt.
!
!  bildnummer
!  zeit
!  i, x(i), y(i), z(i)
!
     2  nkoerp
     1017.196       913.6613       1021.882        mxyz
     5
   1.0000000E-01 zeit
     1    15.92459      -52.21842       29.78373
     2    15.78367      -52.43213       29.33962
     6
   1.2000000E-01 zeit
     1    15.88788      -52.19111       29.74960
     2    15.74572      -52.45454       29.29876
   usw.
```

7. Literatur

Literatur zu DLT

[1] ABDEL-AZIZ, Y.I., KARARA, H.M.: *Direct linear transformation from comparator co-ordinates into space co-ordinates in close range photogrammetry*. Proceedings of the Symposium on Close-Range-Photogrammetry, American Society of Photogrammetry, Falls Church, Virginia, 1-18, (1971)

[2] BALL, K.A., PIERRYNOWSKI, M.R.: *A modified direct linear transformation (DLT) calibration procedure to improve the accuracy of 3D reconstruction for large volumes*. Biomechanics XI-B, de Groot, G., Hollander, A.P., Huijing, P.A., van Ingen Schenau, G.J., Eds., Free University Press, Amsterdam, 1045-50, (1988)

[3] BOPP, H., KRAUSS, H.: *Extension of the 11-parameter solution for the on-the-job calibrations of non metric cameras*. International Archives of Photogrammetry, 22, 7-12, (1978)

[4] BOPP, H., KRAUSS, H.: *Ein Orientierungs und Kalibrierungsverfahren für nicht-topographische Anwendungen der Photogrammetrie*. AVN, 5, 182-188, (1978)

[5] BOPP, H., KRAUSS, H.: *A simple and rapidly converging orientation and calibration method for non topographic applications*. Proceedings of the American Society of Photogrammetry, American Society of Photogrammetry, Fall technical meeting, Falls Church, 425-32, (1977)

[6] KARARA, H.M.: *Non-metric cameras*. Developments in Close Range Photogrammetry, Atkinson, K.B., Ed., Applied Science Publishers, London, 63-80, (1980)

[7] KARARA, H.M., ABDEL-AZIZ, Y.I.: *Accuracy aspects of non-metric images*. Photogrammetric Engineering, 40, 1017-27, (1974)

[8] KARARA, H.M., MARZAN, C.T.: *A computer program for direct linear transformation solution of the colinearity condition, and some applications of it*. Proceedings of the Symposium on Close-Range Photogrammetric Systems, American Society of Photogrammetry, Falls Church, Virginia 420-449 (1975)

[9] SHAPIRO, R.: *Direct linear transformation method for three-dimensional cinematography*. Research Quarterly, 49, 197-205, (1978)

[10] WOOD, G.A., MARSHALL, R.N.: *The accuracy of DLT extrapolation in three-dimensional film analysis*. Journal of Biomechanics, 19, 781-85, (1986)

[11] YEADON, M.R.: *A method for obtaining three dimensional data on ski jumping using pan and tilt cameras.* International Journal of Sport Biomechanics, 5, 238-47, (1989)

[12] YEADON, M.R.: *The simulation of aerial movement. Part I: The determination of orientation angles from film.* Journal of Biomechanics, 23, 59-66, (1990)

Mathematische Literatur

[13] GOLUB, G.H., VAN LOAN, CH.F.: *Matrix Computations.* Second Edition, The John Hopkins University Press, London, (1990)

[14] SCHWARZ, H.R.: *Numerische Mathematik.* Teubner, Stuttgart, (1986)

Physikalische Literatur

[15] HECHT, E., ZAJAC, A.: *Optics.* Addison Wesley Publishing Company, (1974)

Literatur zur Programmierung

[16] AMERICAN NATIONAL STANDARDS INSTITUTE: *American National Standard Programming Language FORTRAN 90.* New York, NY, Document: ANSI X3.198-1992

[17] AMERICAN NATIONAL STANDARDS INSTITUTE: *American National Standard Programming Language FORTRAN.* New York, NY, Document: ANSI X3.9-1978

[18] AMERICAN NATIONAL STANDARDS INSTITUTE: *American National Standard for Information Systems – Programming Language – C.* New York, NY, Document: ANSI X3.159-1990

[19] ANDERSON, E., BAI, Z., BISHOF, C., DEMMEL, J., DONGARRA, J., DU CROZ, J., GREENBAUM, A., HAMMARLING, S., MCKENNEY, A., OSTROUCHOV, S., SORENSON, D.: *Lapack Users' Guide.* SIAM, Philadelphia (1992)

[20] DONGARRA, J.J., MOLER, C.B., BUNCH, J.R., STEWART, G.,W.: *Linpack Users' Guide.* SIAM, Philadelphia (1979)

[21] GEHRKE, WILHELM: *Fortran 90 — Referenz-Handbuch.* Carl Hanser Verlag, München, (1991)

[22] HAIRER, E.: *Private Mitteilung, Übersendung von Vorlesungsprogrammen.* Université de Genève, Dept. de mathématiques, Case de postale 240, CH-1211 Genève 24, Switzerland, (1991)

[23] KERNIGHAN, B.W., RITCHIE, D.M.: *The C Programming Language, Second Edition: ANSI C.* Prentice-Hall, (1988)

Anhang: Kommentarteil von film3d

```
!
!  written by       martin moessner
!  latest change    1992-06-30
!  email            martin.moessner@uibk.ac.at
!
!
!  this code is written in ansi fortran 90 and contains
!  ansi cpp directives !
!  ( see iso/iec 1539:1991 and
!    ansi - programming language fortran : x3.198-1992 )
!
!  versions in
!     ansi fortran 77
!     ( ansi - programming language fortran : x3.9-1978 )
!  and
!     ansi c ( ansi - programming language c : x3.159-1989 )
!  are available too. the fortran 77 code uses some minor
!  extensions of the standard ( -> names with more than 6
!  characters, underscores '_' )
!
! ------------------------------------------------------------------
!
 subroutine film3d( &
    nmess, nkalib, nbild, nkoerp, lwork, work, liwork, iwork )
 integer nmess, nkalib, nbild, nkoerp, lwork, liwork
 integer, dimension(liwork) :: iwork
 double precision, dimension(lwork) :: work
!
! ------------------------------------------------------------------
!
!  this routine makes a dlt-analysis ( direct linear transfor-
!  mations ) with 11 or 16 parameters of given film-data. the
!  object, whose coordinates should be evaluated, must move
!  through a field of fixed passpoints and is filmed by two
!  cameras ( video, film .. ). to avoid interpolation errors,
!  every film frame should show at least 6 ( resp. 8 for dlt-
!  analysis with 16 parameters ) passpoints, which are not
!  complanar, and uniformly surround the object. error reduction
!  is done by overdetermined linear equations. explicit
!  synchronisation of film-data is possible.
!
!  basic properties of this implementation:
!
```

```
!  calibration is done independently for both cameras and for
!  every frame! it follows:
!
!     - passpoints can change from frame to frame
!     - passpoints can change from camera 1 to camera 2 at the
!       same frame
!
!     - the cameras can be panned and tilted
!     - the camera lenses can be zoomed
!
! ---------------------------------------------------------------
!
! this routine reads following files
!
!    film3d.pp     ( passpoint information )
!    film3d.b1     ( image information of first film )
!    film3d.b2     ( image information of second film )
!    film3d.t1     ( time information of first film )
!    film3d.t2     ( time information of second film )
!
! writes to the files
!
!    film3d.xyz    ( analysis results
!                    time, x-, y-, z-coordinates of body-point per
!                    each frame )
!    film3d.ppn    ( new coordinates for the passpoints )
!
! and optionally writes the file film3d.o_, where _ stands for 1, 2,..
! these files can be used for debugging aids.
!
! ---------------------------------------------------------------
!
!  input parameters:
!
!  nmess  integer
!         is the number of surveyed passpoints
!
!  nkalib integer
!         is the maximum number of calibration passpoints per frame
!
!  nbild  integer
!         is the maximum number of images to process
!
!  nkoerp integer
!         number of processed body-point per each frame
!
!  lwork  integer
```

```
!         dimension of double precision work vector work
!
!         lwork >= 3 * nmess + 2 * nbild + 14 * nkalib +
!                  4 * par * nkalib + 4 * par + 7 * nkoerp + 5
!         par = 11 or 16 ( number of parameters for dlt )
!
!  work   double precision work(lwork)
!         double precision work vector
!
!  liwork integer
!         dimension of integer work vector iwork
!
!         liwork >= 2 * nbild + 2 * nkalib + par + 5
!
!  iwork  integer iwork(liwork)
!         integer work vektor
!
! ---------------------------------------------------------------
!
! special tuning of the code
!
! following parameters are set to default values if the corresponding
! entries of iwork resp. work are zero
!
! iout    integer - iwork(1)
!         controls additional output generation
!         iout <= 0   no additional output
!         iout > 0    for n = min ( iout, 6 ) the files
!                     film3d.o1, film3d.o2, .. , film3d.on
!                     are generated
!                     ( default 0 )
!
! par     integer - iwork(2)
!         number of parameters of the linear direct transform
!         ( default 11 )
!
! scale   integer - iwork(3)
!         controls internal scaling
!         scale == 0   automatically scaling of image- and
!                      object-coordinates
!         scale == 1   image-coordinates are scaled
!         scale == 2   object-coordinates are scaled
!         scale == 3   no scaling
!         ( default 0 )
!
! itmax   integer - iwork(4)
!         maximum number of iterations for 16 parameters
```

```
!         ( default 10 )
!
! bmin,   double precision - work(1)
! bmax    double precision - work(2)
!         if image-scaling is on, image coordinates between bmin
!         and bmax are mapped to -1, +1
!
! error estimations:
!
! res1    double precision - work(3)
! res2    double precision - work(4)
! resm    double precision - work(5)
! ires    integer - iwork(5)
!
!         error estimators in 1, 2 and infinity norm
!         these estimators are get by processing passpoints which
!         are visible to both cameras. ires is the number of the
!         used passpoints
!
! ------------------------------------------------------------------
!
! description of the input files:
!
! film3d.pp ( passpoint field )
! ============================
!
!    9 lines of comment
!    for every passpoint a line i, j, x, y, z, where
!
!       i   integer
!           is not used
!       j   integer
!           the number of the passpoint
!       x   float
!           the x coordinate of passpoint j
!       y   float
!           the y coordinate of passpoint j
!       z   float
!           the z coordinate of passpoint j
!
!    it is assumed that x, y, z are cartesian coordinates.
!
! film3d.b1, film3d.b2 ( image data )
! ==================================
!
!    10 lines of comment
!
```

```
!     bf
!     bl
!     frame
!     frame
!      ...
!
!     it is
!
!     bf   integer
!          number of first image
!     bl   integer
!          number of last image
!     frame
!          this is a block of data, which describes an image
!          it builds up from
!
!          nbild
!          0
!          ptr_1
!          ptr_2
!          ...
!          ptr_n
!          0
!          x_p_1
!          y_p_1
!          x_p_2
!          y_p_2
!          ...
!          x_p_n
!          y_p_n
!          0
!          x_k_1
!          y_k_1
!          ...
!          x_k_nkoerp
!          y_k_nkoerp
!
!          here is
!
!          nbild   integer
!                  number of image
!          0       the number 0
!          ptr_1, .. , ptr_n    integer
!                  ptr_i are the pointers for the passpoints
!                  if ptr_i = j the j-th passpoint is used
!                  ( see film3d.pp )
!                  the first 0 ends the pointer-field and determines n
```

```
!         x_p_1, y_p_1, .. , x_p_n, y_p_n   float
!                 image-coordinates of the passpoints
!                 x_p_i, y_p_i are the x and y coordinate of
!                 passpoint j, where j = ptr_i
!         x_k_1, y_k_1, .., x_k_nkoerp, y_k_nkoerp
!                 image-coordinates of the body points, which should
!                 be processed
!                 x_k_i, y_k_i are the x and the y coordinates of
!                 the body point i. note nkoerp is known to the code
!                 via argument list of film3d
!
! film3d.t1, film3d.t2 ( time data )
! ==================================
!
!    aequidistant frames - write following 5 lines into the file:
!
!       1 ! 1 for aequidistant images
!         ! 2 for special film-time evaluation
!       t
!       i
!       d
!
!    here is
!
!       t   real
!           time of first image
!       i   integer
!           number of images
!       d   real
!           time gap between two succesive images
!
!    not aequidistant frames - no documentation
!
! ------------------------------------------------------------------
!
! internal quantities:
!
! par     integer
!         number of parameters for dlt transform
!         ( only 11 and 16 supported )
! passmin integer
!         minimum numbers of passpoints for dlt transform
!         passmin = 6 for par = 11 and passmin = 8 for par = 16
! scale   integer
!         controls scaling
! b_min, b_max double precision
!         threshold values for image-scaling
```

```
! t_1, t_2 double precision, dimension(nbild)
!         t_j(i) is the time of the i-th fram of camera j
! lt_1, lt_2 integer
!         lt_j is the runtime dimension of t_j ( lt_j <= nbild )
! freq_1, freq_2 double precision
!         estimation of the time between two frames
!         ( note: ordinary films do not have constant transport
!           velocities )
! syn     integer, dimension(2,nbild)
!         syn(1,j) and syn(2,j) are synchronized film-images
! lsyn    integer
!         runtime dimension of syn ( lsyn <= nbild )
! mxyz    double precision, dimension(3)
!         center of mass of the passpoints
! nb1, nb2 integer
!         frame numbers
! pass_1, pass_2, pass integer
!         number of passpoints for the current image
! ptr_1, ptr_2, ptr integer, dimension(nkalib)
!         pointers for the passpoints of the current image
! port_1, port_2, port double precision, dimension(3,nkalib)
!         space-coordinates of the passpoints for the current image
! pbld_1, pbld_2, pbld, double precision, dimension(2,nkalib)
!         image-coordinates of the passpoints for the current image
! mitt    double precision, dimension(3)
!         center of mass the passpoint for the current image
! kort    double precision, dimension(3,nkoerp)
!         space-coordinates of the evaluated object points
! kbld_1, kbld_2, kbld double precision, dimension(2,nkoerp)
!         image-coordinates of the object point
! a, aa   double precision, dimension(n2kal,par)
!         matrices for dlt transform
! c, cc   double precision, dimension(n2kal)
!         right hand sides for dlt transform
!
! nt1, nt1 integer
!         unit numbers of film3d.t1 and film3d.t2
! nk1, nk2 integer
!         unit numbers of film3d.b1 and film3d.b2
! npass, npass1 integer
!         unit numbers of film3d.pp and film3d.ppn
! nxyz    integer
!         unit number of film3d.xyz
!
! ---------------------------------------------------------------
!
!  literature:
```

```
!
!     moessner, martin
!        3d-filmauswertung bei mitgeschwenkter und gezoomter kamera.
!        institutsbericht, inst. fuer mathematik und geometrie,
!        technikerstr. 13, innsbruck, austria, august (1992)
!
!     marzan, c.,t., karara, h.,m.
!        a computer program for direct linear transformation
!        solution of the colinearity condition, and some
!        applications on it. proceedings of the symposium on
!        close-range photogrammetric systems, american society
!        of photogrammetry, falls church, virginia 420-476 (1975)
!
! ------------------------------------------------------------------
!
!  this work has been supported by
!  the austrian research foundation
!
! ------------------------------------------------------------------
!
 integer is, ip1, ip2, iip, iie
 integer im, it1, it2, ipo1, ipo2, ipb1, ipb2, ikb1, ikb2, iko, &
    ib1, ib2, ibb, iba, ia, iaa, ic, icc, ie
 integer iout, n2kal, par, passmin, scale, itmax
 double precision bmin, bmax
!
!  check input parameters
!
 iout = max( iwork(1), 0 )
!
 par = 11
 if ( iwork(2) == 16 ) then
    par = 16
 else
    if ( par /= 11 ) then
       write(0,'(/,1x,a,/)') 'film3d: wrong input par. set to 11'
    end if
 end if
 passmin = 6; if ( par == 16 ) passmin = 8
!
 scale = iwork(3)
 if ( ( scale < 0 ) .or. ( scale > 3 ) ) then
    write( 0,'(/,1x,a,/)') 'film3d: wrong input scale. set to 0'
    scale = 0
 end if
!
 if ( iwork(4) <= 0 ) then
```

```
      itmax = 10
   else
      itmax = iwork(4)
   end if
!
   bmin = work(1)
   bmax = work(2)
   if ( ( work(1) == 0 ) .and. ( work(2) == 0 ) ) bmax = 4000
!
   n2kal = 2 * nkalib
!
   im = 6; it1 = im + 3 * nmess; it2 = it1 + nbild
   ipo1 = it2 + nbild; ipo2 = ipo1 + 3 * nkalib
   ipb1 = ipo2 + 3 * nkalib; ipb2 = ipb1 + 2 * nkalib
   ikb1 = ipb2 + 2 * nkalib; ikb2 = ikb1 + 2 * nkoerp
   iko = ikb2 + 2 * nkoerp
   ib1 = iko + 3 * nkoerp; ib2 = ib1 + par; ibb = ib2 + par
   iba = ibb + par
   ia = iba + par; iaa = ia + par * n2kal
   ic = iaa + par * n2kal; icc = ic + n2kal; ie = icc + n2kal - 1
   if ( ie > lwork ) then
      write(0,'(/,2x,a,2i6)') &
         'film3d: lwork too small', ie, lwork
      stop
   end if
   is = 6; ip1 = is + 2 * nbild; ip2 = ip1 + nkalib
   iip = ip2 + nkalib; iie = iip + par - 1
   if ( iie > liwork ) then
      write(0,'(/,2x,a,2i6)') &
         'film3d: liwork too small', iie, liwork
      stop
   end if
   call filmcor3d( &
      par, passmin, scale, itmax, bmin, bmax, &
      nmess, nkalib, n2kal, nbild, nkoerp, &
      iwork(is), iwork(ip1), iwork(ip2), iwork(iip), &
      work(im), work(it1), work(it2), &
      work(ipo1), work(ipo2), work(ipb1), work(ipb2), &
      work(ikb1), work(ikb2), work(iko), &
      work(ib1), work(ib2), work(ibb), work(iba), &
      work(ia), work(iaa), work(ic), work(icc), iout, &
      work(3), work(4), work(5), iwork(5) )
   end
!
! ---------------------------------------------------------------------
!
   subroutine filmcor3d( &
```

```
     par, passmin, scale, itmax,  bmin, bmax, &
     nmess, nkalib, n2kal, nbild, nkoerp, &
     syn, ptr_1, ptr_2, ip, &
     mess, t_1, t_2, &
     port_1, port_2, pbld_1, pbld_2, &
     kbld_1, kbld_2, kort, &
     b_1, b_2, bb, alph, &
     a, aa, c, cc, iout, &
     res1, res2, resm, ires )
  integer par, passmin, scale, itmax, ires, &
     nmess, nkalib, n2kal, nbild, nkoerp, iout
  double precision bmin, bmax, res1, res2, resm
  integer, dimension(2,nbild) :: syn
  integer, dimension(nkalib) :: ptr_1, ptr_2
  integer, dimension(par) :: ip
  double precision, dimension(3,nmess) :: mess
  double precision, dimension(nbild) :: t_1, t_2
  double precision, dimension(3,nkalib) :: port_1, port_2
  double precision, dimension(2,nkalib) :: pbld_1, pbld_2
  double precision, dimension(2,nkoerp) :: kbld_1, kbld_2
  double precision, dimension(3,nkoerp) :: kort
  double precision, dimension(par) :: b_1, b_2, bb, alph
  double precision, dimension(n2kal,par) :: a, aa
  double precision, dimension(n2kal) :: c, cc
!
```

Sprungschanze

GSCHWANDTKOPF

VERMESSUNG	ZEICHNUNG	SACHBEARBEITER
19–03–92	22–04–92	Ing. Thomas Alt

VERMESSUNG G R I M M

A – 6020 INNSBRUCK – TECHNIKERSTRASSE 3

TELEFON : 0512 / 2 88 88 2
MOBILTELEFON : 0663 / 85 82 62
TELEFAX : 0512 / 2 88 1 44

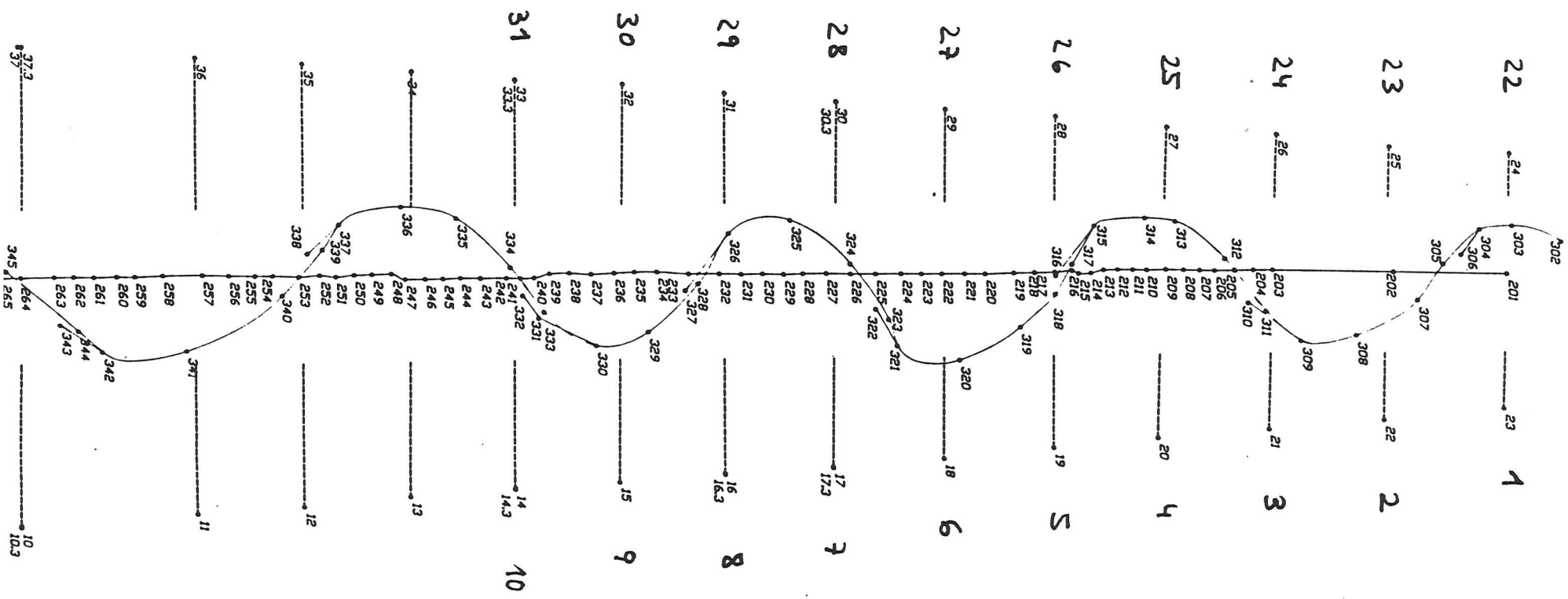

Abb. 3: Anordnung der Paßpunkte

von Hand eingetragene Zahlen sind Nummern der Bildauswertung