Optischer Fluss und Image Mosaicing nach Kourogi

Wolfgang Konen
Institut für Informatik, FH Köln

22.01.2009

Abstract. Dieser kurze Technical Report enthält einige Anmerkungen zum Optischen Fluss und Erläuterungen zum Algorithmus nach [Kourogi99], der den Optischen Fluss zur Erstellung von Übersichtsbildern (Panorama, Image Mosaic) aus einer Videosequenz nutzt.

Was ist Optischer Fluss?

Nehmen wir an, dass wir uns in einer statischen Welt befinden.

Bewegt sich ein Beobachter (eine Kamera) durch diese statische Welt, so entsteht auf seiner Retina (auf dem Kameratarget) dennoch Bewegung, der sog. "Optische Fluss" (engl. *optical flow*). Abbildung 1 links zeigt ein Beispiel für den Optischen Fluss aus der realen Welt.

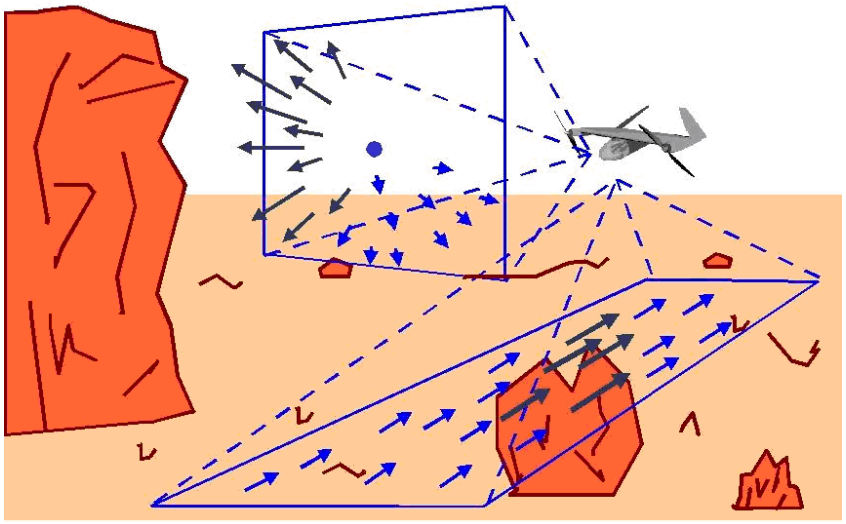

Abbildung 1: Optischer Fluss (O.F.), den ein Vogel, Insekt oder ein Flugzeug sieht. Der blaue Punkt vorne ist der FOE (focus of expansion), rechts davon ist der O.F. größer aufgrund des nahen Felsens.

Mathematischer versteht man unter dem optischen Fluss das Verschiebungsvektorfeld (engl. *motion field*) $\begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix}$, das für jeden Punkt der Welt angibt, welche Verschiebung auf der Bildebene er in einem Zeitintervall Δt erfährt. Abbildung 2 zeigt zwei Beispiele für optischen Fluss: Links der optische Fluss, wenn man sich nach hinten bewegt, denn dann wandern alle Punkte der statischen Welt einwärts. Rechts der optische Fluss, wenn man den Kopf nach rechts dreht, denn dann wandert alle Punkte der statischen Welt nach links.

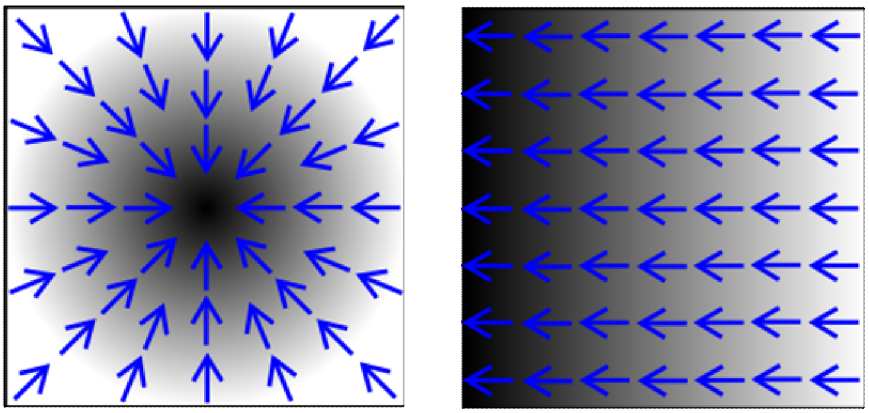

Abbildung 2: Zwei Beispiele für optischen Fluss. Dargestellt sind einige Vektoren (Pfeile) aus einem kontinuierlichen Verschiebungsvektorfeld.

Es sei $\begin{pmatrix} u \\ v \end{pmatrix}$ der wahre Verschiebungsvektor für Pixel (x,y). Eine übliche Annahme der Computer Vision ist nun, dass sich die Lichtstärke, die von einem Punkt der statischen Welt ausgeht, im Zeitintervall Δt nicht ändert. (Wir sehen also von Glühwürmchen, Taschenlampen und Pulsaren für den Moment ab. Aber auch von winkelabhängigen Reflektanzeigenschaften.) Dann können wir schreiben

(1) $$I(x+u, y+v, t) - I(x, y, t-\Delta t) = 0$$

Diese Gleichung bedeutet einfach: Die Lichtstärke, die vorher, also zum Zeitpunkt *t-Δt,* im Bildpunkt *(x,y)* war, befindet sich zum Zeitpunkt *t* im Bildpunkt *(x+u,y+v)*.

Wenn wir annehmen, dass die Intensität I eine stetige Funktion ist – eine Annahme, die in der Realität wegen Sprüngen im Grauwertverlauf leider öfters *nicht* zutrifft – dann können wir Gl. (1) für kleines *Δt*, mithin auch kleines u und v, in eine Reihe entwickeln (linearisieren) und erhalten

$$I(x+u, y+v, t) - I(x, y, t-\Delta t) = 0$$
$$I(x, y, t) + I_x u + I_y v - \left(I(x, y, t) - I_t \Delta t\right) = 0$$

(2) $$\boxed{I_x u + I_y v + I_t \Delta t = 0}$$

Hierbei sind I_x, I_y und I_t die partiellen Ableitungen von I nach x, y und t. Gl. (2) ist die berühmte, von Horn und Schunck [HornSchunck81] entwickelte Gleichung des optischen Flusses. Wenn wir auf das Zeitinkrement $\Delta t=1$ spezialisieren (von einem Videoframe zum nächsten) und eine andere Schreibweise für die partielle Ableitung verwenden, so erhalten wir als äquivalente Variante eine Form, die sich oft in der Literatur (so auch bei [Kourogi99]) findet:

(2') $$\boxed{\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} = 0}$$

Der optische Fluss spielt auch eine große Rolle bei der Navigation von mit Sehsinn ausgestatteten Objekten (Insekten, unbemannte Flugkörper, Robotik)

Die Methode nach Kourogi'99

Ziel: Gegeben seien zwei aufeinander folgende Frames aus einem Video. Wie kann man das Verschiebungsvektorfeld zwischen ihnen schätzen, und das möglichst in Echtzeit?

Es sei wiederum $\begin{pmatrix} u \\ v \end{pmatrix}$ der wahre Verschiebungsvektor für Pixel *(x,y)* bei $\Delta t=1$. D. h. es gelte[1]

$$I(x+u, y+v, t) - I(x, y, t-1) = 0$$

Leider kennen wir den wahren Verschiebungsvektor in der Regel nicht! Was tun? – Die berühmte Gl. (2) hat einen bekannten, entscheidenden Nachteil: Sie ist ***allein*** und ist so nicht hinreichend dafür, die ***zwei*** Unbekannten u und v zu bestimmen.

Pseudo Motion

[Kourogi99] geht einen auf den ersten Blick brutal vereinfacht anmutenden Weg:

- Wenn wir u ausrechnen wollen, dann setzen wir in Gl. (2) einfach den v-Term zu Null
- Wenn wir v ausrechnen wollen, dann setzen wir in Gl. (2) einfach den u-Term zu Null

Dies führt auf die sog. **Pseudo Motion**:

(3) $$\begin{pmatrix} u_p \\ v_p \end{pmatrix} = \begin{pmatrix} -I_t / I_x \\ -I_t / I_y \end{pmatrix},$$

Hierbei sind I_x, I_y und I_t die partiellen Ableitungen von I nach x, y und t, die durch ihre diskreten numerischen Vertreter (s.u., Implementierungsaspekte) anzunähern sind. Die Gl. (3) darf natürlich nur dort benutzt werden, wo I_x bzw. I_y nicht Null ist. Dieser Pseudo Motion Vektor ist zwar oftmals ungenau, aber nimmt man viele Samples so sollte, unter recht allgemeinen Annahmen, die stochastische Größe $u_p - u$ um den Mittelwert 0 streuen. Weil aber bei der Pseudo Motion oft auch 'wirre' Werte vorkommen werden, führt [Kourogi99] folgenden **Akzeptanztest** durch:

(4) $$\left| I(x+u_p, y+v_p, t) - I(x, y, t-1) \right| < T,$$

wobei T eine geeignet zu wählende Grauwertschwelle ist, z.B. $T=5$. Nur Pseudo Motion Vektoren, die diesen Test passieren, werden zur weiteren Analyse herangezogen.

Die Sache mit der Pseudo Motion hat also den Vorteil, dass wir sehr schnell Werte für u und v ausrechnen können. Sie hat den Nachteil, dass sie manchmal funktioniert, aber oft auch nicht funktioniert. Bevor wir weitergehen, machen wir uns erst einmal anschaulich klar, wann es funktionieren kann und wann es schief gehen muss:

[1] In [Kourogi99] wird das Frame zum Zeitpunkt t wird auch als "current" oder I_c, das zum Zeitpunkt t-1 auch als "reference" oder I_r bezeichnet.

Wann ist der Pseudo Motion Vektor genau? – Wenn vom Zeitpunkt t-1 bis t eine **rein lineare** Grauwertrampe am Beobachtungsort (x,y) vorbeiwandert. Nehmen wir als Beispiel eine 1-dimensionale (1D) Bewegung in x, bei der eine Grauwertrampe [130 132 134 136 138] mit einer Geschwindigkeit von –3 Pixel/Frame am Beobachtungsort vorbeiwandert. Zum Zeitpunkt t-1 sei der Grauwert am Beobachtungsort 132, dann ist er also 138 zum Zeitpunkt t am Beobachtungsort. Wir errechnen $I_t = 138–132 = 6$ und $I_x = (134–132)/1 = 2$ und Gl. (3) führt richtigerweise auf $u_p = –6/2 = –3$.

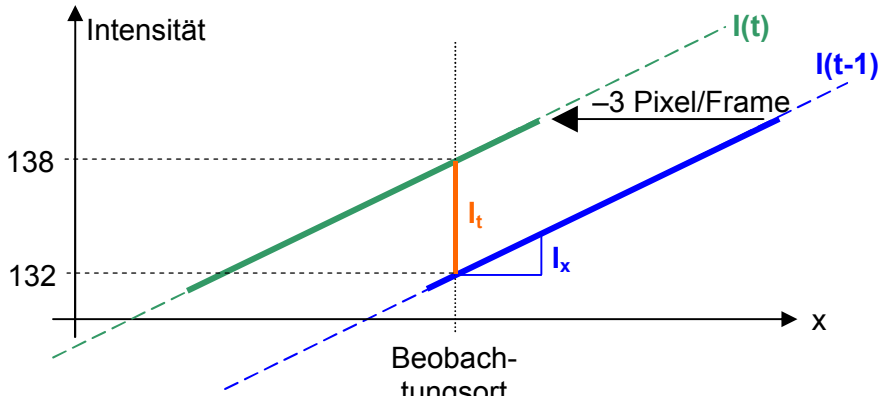

Abbildung 3: Die Pseudo Motion $–I_t/I_x$ ist bei linearer Grauwertrampe identisch zum wahren Verschiebungsvektor. (Gilt so nur im 1D-Fall)

Compensated Motion

Wir sehen umgekehrt aus **Abbildung 3**, wann der Pseudo Motion Vektor falsch sein muss:

- Wenn der Grauwertbereich, der von t-1 bis t am Beobachtungsort (x,y) vorbeiwandert, eine Sprungstelle (Diskontinuität, engl. *discontinuity*) enthält,
  - Das wird bei Bildern häufig der Fall sein. Dann hat der zeitliche Verlauf in I_t Änderungen "gesehen", die der lokale Gradient, also die Ableitung nach x, nicht kennen kann.
  - Die Wahrscheinlichkeit für Sprungstellen wird umso größer, je größer die Verschiebungsvektoren sind
- Zwar keine unstetigen, aber nicht-lineare Grauwertverläufe (z.B. quadratisch). Der Effekt wird natürlich umso stärker, je größer der Motion Vektor ist.

In beiden Fällen kann die sog. Compensated Motion [Kourogi99] helfen. Die Idee dahinter ist die folgende: Wenn wir aus irgendwelchen Quellen eine – wenn auch nur grobe – Schätzung $\begin{pmatrix} u_c \\ v_c \end{pmatrix}$ für das Verschiebungsvektorfeld haben, dann können wir diese Verschiebungskomponente schon einmal überall kompensieren. Der verbleibende Rest $\begin{pmatrix} u_r \\ v_r \end{pmatrix}$ ist dann (hoffentlich!) kleiner als das gesamte $\begin{pmatrix} u \\ v \end{pmatrix}$, und vielleicht gilt für diesen Rest die lineare Näherung.

Abbildung 4 zeigt ein Beispiel für Diskontinuitäten:

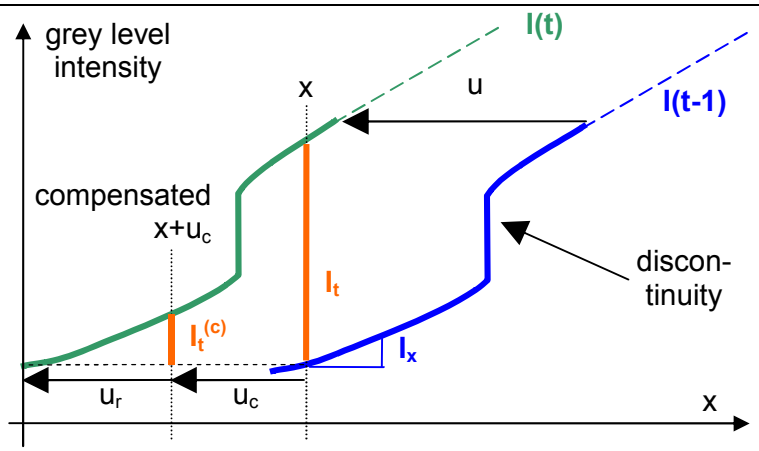

Abbildung 4: Zur Funktionsweise der Compensated Motion

Erläuterung: Wir stehen im Beobachtungsort x und beobachten, wie die "Welt" in Form eines hier 1-dimensionalen Grauwertverlaufes an uns vorbeiwandert. Im Zeitraum von t-1 bis t wandert eine Diskontinuität am Beobachtungsort x vorbei. Die Ableitungen I_t und I_x passen dann nicht mehr zusammen: Im Beispiel ist der Quotient $-I_t/I_x$ viel zu groß. Wenn wir aber eine halbwegs brauchbare Schätzung u_c für die Verschiebung haben, dann können wir uns an den kompensierten Ort $x+u_c$ begeben. Dann umfasst die Differenz

$$I_t^{(c)} = I(x+u_c,t) - I(x,t\text{-}1)$$

nicht mehr die Diskontinuität, der Quotient $-I_t^{(c)}/I_x$ liegt viel näher an der Verschiebung u_r (s. Bild), mithin liegt der Term $-I_t^{(c)}/I_x + u_c$ viel näher am wahren Verschiebungsvektor u.

LS-Schätzung der affinen Parameter

[Die nachfolgende Betrachtung wurde im Grunde schon im Vortrag "Warping" behandelt, sie ist hier nur der Vollständigkeit halber noch einmal aufgeführt.]

Gegeben ein Pseudo-Motion-Feld (u_p,v_p), welche affinen Motion-Parameter beschreiben es bestmöglich? – Die Antwort zerfällt in zwei entkoppelte Teilprobleme: Suche Parameter (a_1,a_2,a_3) mit

$$\left[u_c(a_1,a_2,a_3) - u_{pi}\right]^2 = \left[a_1x_i + a_2y_i + a_3 - u_{pi}\right]^2 = \text{Min.}$$

bzw. suche Parameter (a_4,a_5,a_6) mit

$$\left[v_c(a_4,a_5,a_6) - v_{pi}\right]^2 = \left[a_4x_i + a_5y_i + a_6 - v_{pi}\right]^2 = \text{Min.}$$

wobei i=1..N über alle Pixel läuft, die den Akzeptanztest bestanden haben. Wir behandeln nachfolgend o.B.d.A. nur das u_p-Problem. Es gibt i.a. viel mehr Gleichungen als Unbekannte, d.h. das Gleichungssystem lässt sich nur im LS-Sinne lösen:

$$(\mathbf{A}\cdot\mathbf{a}-\mathbf{b})^2=\text{Min.}$$

(5) mit $\mathbf{A}=\begin{pmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_N & y_N & 1 \end{pmatrix}$, $\mathbf{a}=\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$, $\mathbf{b}=\begin{pmatrix} u_{p1} \\ \vdots \\ u_{pN} \end{pmatrix}$

Man beachte, dass **A** und **b** i.d.R. viele Tausend Zeilen lang sein werden! Ein viel einfacheres, nämlich nur 3x3-großes Problem ergibt sich, wenn wir in **Aa=b** beide Seiten von links mit $\mathbf{A}^T$ durchmultiplizieren:

$$(\mathbf{M}\cdot\mathbf{a}-\mathbf{c})^2=\text{Min.}$$

(6) mit $\mathbf{M}=\mathbf{A}^T\mathbf{A}=\begin{pmatrix} \sum x_i x_i & \sum x_i y_i & \sum x_i \\ \sum x_i y_i & \sum y_i y_i & \sum y_i \\ \sum x_i & \sum y_i & \sum 1 \end{pmatrix}$, $\mathbf{c}=\mathbf{A}^T\mathbf{b}=\begin{pmatrix} \sum x_i u_{pi} \\ \sum y_i u_{pi} \\ \sum u_{pi} \end{pmatrix}$

Dabei laufen die Summen über alle i=1..N, die den (jeweils aktuellen) Akzeptanztest bestehen. Deshalb kann sich die Matrix **M** von Iteration zu Iteration ändern, man kann sie (leider!) nicht einmal nur zu Beginn rechnen.

Beide Probleme lassen sich mit der SVD-Inversen zu **A** bzw. **M** (oder in MATLAB mit dem Backslash-Operator) (JAMA: `M.solve(Matrix c)`) lösen.

Wie also bekommen wir eine (grobe) Schätzung $\begin{pmatrix} u_c \\ v_c \end{pmatrix}$? – In der Videoanwendigung evtl. von vorherigem Frame-Paar. Wenn es nur das aktuelle Frame-Paar gibt, dann starten wir mit $\begin{pmatrix} u_c \\ v_c \end{pmatrix}=\begin{pmatrix} 0 \\ 0 \end{pmatrix}$. Wir nehmen, ähnlich wie beim Warping, an, dass das ganze Verschiebungsfeld aus einer bestimmten Klasse von Transformationen stammt, z.B. "affin" oder "Translation". Aus der Pseudo Motion wird eine globale Compensated Motion geschätzt: Welche affinen Parameter minimieren

(7) $$\left\| \begin{pmatrix} u_c(a_1,a_2,a_3) \\ v_c(a_4,a_5,a_6) \end{pmatrix} - \begin{pmatrix} u_p \\ v_p \end{pmatrix} \right\| = \left\| \begin{pmatrix} a_1 x + a_2 y + a_3 \\ a_4 x + a_5 y + a_6 \end{pmatrix} - \begin{pmatrix} u_p \\ v_p \end{pmatrix} \right\|$$

Im Frame zum Zeitpunkt t kann nun die geschätzte Bewegung kompensiert werden. Dies führt auf eine andere partielle Ableitung nach der Zeit (I_x und I_y ändern sich nicht, da sie im Frame t-1 berechnet werden) und eine modifizierte Pseudo Motion Berechnung

$$I_t^{(c)} = I(x+u_c, y+v_c, t) - I(x,y,t-1)$$

(8) $$\begin{pmatrix} u_p \\ v_p \end{pmatrix} = \begin{pmatrix} -I_t^{(c)} / I_x \\ -I_t^{(c)} / I_y \end{pmatrix} + \begin{pmatrix} u_c \\ v_c \end{pmatrix}$$

Das neue (u_p, v_p) wird wieder dem Gültigkeitstest nach Gl. (4) unterworfen. Das Verfahren der wechselnden Berechnung von (u_c, v_c) und (u_p, v_p) wird nun iteriert und führt zu einer verbesserten Schätzung der Motion-Vektoren.

Im Ergebnis erhalten wir ein „dichtes“ Vektorfeld (u_c, v_c).

Tabelle 3: Iteratives Verfahren mit Compensated Motion. ($\bar{u}_p, \bar{v}_p$) ist die Mittelung über alle (u_p, v_p) nach Gl. (8). Angaben in Pixeln. „% accept“ ist der Prozentsatz der Pixel die den Akzeptanztest nach Gl. (4) mit T=5 bestehen.

T=5		wahre Translation		mit Compensated Motion		
		u	v	$\bar{u}_p$	$\bar{v}_p$	% accept
Iteration	1	-10.5	7.6	-1.5 ± 10.0	2.2 ± 12.3	14%
	2	-10.5	7.6	-3.1 ± 8.6	3.8 ± 10.3	16%
	5	-10.5	7.6	-7.4 ± 5.3	7.0 ± 6.8	23%
	10	-10.5	7.6	-10.47 ± 1.2	7.61 ± 1.27	55%

(Die fehlenden Iterationen wurden gerechnet, nur in der Tabelle ausgelassen.)
Man sieht, dass es so auch für große Translationen geht! Recht schnell erzielt man Konvergenz und das mit Subpixelgenauigkeit. Im Laufe der Iterationen erfüllen sehr viel mehr Pixel den Akzeptanztest nach Gl. (4) als es zu Anfang der Fall war.

Literatur

[Kourogi99] http://citeseer.ist.psu.edu/253440.html: M. Kourogi, T. Kurata, J. Hoshino, and Y. Muraoka. *Real-time image mosaicing from a video sequence*. In Proc. ICIP99,vol. 4, 133--137, 1999.

[HornSchunck81] B. K. P. Horn and B. G. Schunck, *Determining optical flow*, Artificial Intelligence, **17**, 1-3, pp. 185--203, 1981.

[KonBS07] W. Konen, B. Breiderhoff, M. Scholz: Real-time image mosaic for endoscopic video sequences, Proc. BVM (Bildverarbeitung für die Medizin), München, 2007.

[Barrows] www.centeye.com/pages/techres/opticflow.html G. L. Barrows erklärt die Grundprinzipien des Optical Flow gut in Bildern, Bedeutung für die Navigation bei Insekten (Biomimetik)