Seminar Robotik: Visual Odometry

Matthias Nieuwenhuisen

07.05.2007

Zusammenfassung

In dieser Seminararbeit stelle ich einen Verfahren zur Rekonstruktion von Trajektorien aus Videodaten in Echtzeit vor, das von David Nistér, Oleg Naroditsky und James Bergen im Paper "Visual Odometry" [5] beschrieben wurde.

1 Einleitung

In diesem Seminar stelle ich ein Verfahren vor, das aus während einer Bewegung aufgenommenen Videodaten in Echtzeit einen Pfad im Raum bestimmt. Dieses Verfahren wird zur Navigation von autonomen Fahr- oder Flugzeugen genutzt, da es im Gegensatz zum Dead Reckoning eine wesentlich genauere Standortbestimmung ermöglicht, ohne auf externe Hilfsmittel wie zum Beispiel künstliche Landmarken oder GPS angewiesen zu sein.

Zur visuellen Navigation sind folgende Schritte nötig:

- Suchen von Orientierungspunkten
- Verfolgen der Positionsänderung der Orientierungspunkte
- Rekonstruktion der eigenen Bewegung

Ich werde in den folgenden Abschnitten die nötigen Schritte, in der Reihenfolge, in der sie vom Algorithmus abgearbeitet werden, im Detail vorstellen. Da bei der Echtzeitverarbeitung von Videodaten pro Bild nur ein sehr kleines Zeitfenster zur Verfügung steht, gehe ich bei einigen teuren Punkten auf Möglichkeiten zur effizienten Implementierung ein. Durch Optimierung aller Schritte wurden in Tests Bildraten von 26 Frames pro Sekunde auf Standardhardware erreicht.

2 Featureerkennung

Wie bei jedem sensorbasierten Lokalisierungsverfahren ist es nötig Referenzpunkte zur Orientierung zur Verfügung zu haben. Da bei diesem Verfahren auf Kamerabilder zurückgegriffen wird, müssen erst mögliche Referenzpunkte aus den Bildern extrahiert werden. Für den Algorithmus ist es wichtig, dass diese Punkte über mehrere Bilder zur Verfügung stehen, um eine relative Positionsänderung feststellen zu können.

Hierzu werden zuerst in allen Bildern Merkmale (Features) gesucht, die später korreliert werden können. Es müssen also Features gesucht werden, die möglichst über alle Bilder gleich bleiben. In diesem Fall bieten sich "Harris Corners" [2] an, da diese bei nur leicht zueinander verschobenen Bildern sehr gut die selben Stellen kennzeichnen.

Anschaulich sind Harris Corner Intensitätsveränderungen in einem kleinen Fenster, dass man über das Bild schiebt. Gibt es in keine Richtung eine Veränderung, befindet man sich über einer leeren Fläche. Gibt es nur in eine Richtung keine Änderung, befindet man sich über einer Kante. Sind hingegen in alle Richtungen Änderungen feststellbar, befindet das Fenster sich über einer Ecke.

Mathematisch lässt sich dieses Fenster ausdrücken durch eine Funktion

$$E_{x,y} = \sum_{u,v} w_{u,v} \left[I_{x+u,y+v} - I_{u,v} \right]^2, \tag{1}$$

wobei $I_{u,v}$ die Ableitung des Bildes an der Stelle (u, v) bezeichnet und $w_{u,v}$ angibt, ob sich (u, v) innerhalb oder ausserhalb des Fensters befindet. Für kleine Verschiebungen des Fensters lässt sich dies approximieren zu

$$E_{x,y} = (x, y)\, M\, (x, y)^T \tag{2}$$

$$M = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2. \end{pmatrix} \tag{3}$$

Als Qualitätsmaß für die Corner Response gilt nun die Funktion

$$r = \det(M) - k\,\mathrm{trace}(M)^2. \tag{4}$$

Als Feature werden diejenigen Punkte übernommen, für die kein anderer Punkt in der Nachbarschaft (hier: ein 5×5 Fenster) eine stärkere Corner Response r hat (für das vorgestellte Verfahren wurde $k = 0.06$ gewählt).

Um diese Berechnungen effizient durchzuführen, das heisst unter optimaler Cacheausnutzung, werden die Frames jeweils Zeilenweise verarbeitet. Da mit Ableitungen gerechnet wird, müssen auch Zeilen vor und hinter der aktuellen Zeile

betrachtet werden. Um unnötige Neuberechnungen zu vermeiden, wird ein Ringpuffer über fünf Zeilen verwendet um Zwischenergebnisse zu speichern. Nacheinander werden für jede Zeile vier Sweeps durchgeführt, die die Ableitungen, die Matrix M, sowie deren Determinante und Spur berechnen. Weiterhin wird in dem 5×5 Fenster zu einer Corner die Suche sofort abgebrochen und die Corner verworfen, sobald die erste stärkere Corner Response entdeckt wurde.

3 Feature Matching

Nachdem die Features eines Bildes extrahiert wurden, müssen die jeweils korrespondierenden Features in den anderen Frames gefunden werden. Hierzu wird jedes Feature des aktuellen Frames gegen alle Features bis zu einem bestimmten Abstand des anderen Frames getestet. Für jedes Featurepaar wird die normalisierte Korrelation über die Intensitäten in einem 11x11 Pixelfeld, mit dem Feature als Mittelpunkt, berechnet:

$$r = \frac{n \sum I_1 I_2 - \sum I_1 \sum I_2}{\sqrt{n \sum I_1^2 - \left(\sum I_1\right)^2} \sqrt{n \sum I_2^2 - \left(\sum I_2\right)^2}} \tag{5}$$

Da ein Feature zwar in vielen Korrelationen vorkommen kann, jedoch immer das selbe 11x11 Pixelfeld hat, muss der Korrelationsterm nicht in jedem Schritt komplett berechnet werden, sondern kann in Teilen vorberechnet werden:

$$A = \sum I \tag{6}$$

$$B = \sum I^2 \tag{7}$$

$$C = \frac{1}{\sqrt{nB - A^2}} \tag{8}$$

Der Korrelationsterm, der in jedem Schritt berechnet werde muss, vereinfacht sich so zu:

$$\left(n \sum I_1 I_2 - A_1 A_2\right) C_1 C_2 \tag{9}$$

Dabei kann das Skalarprodukt $\sum I_1 I_2$ mit MMX Instruktionen sehr effizient berechnet werden.
Als bevorzugter Korrelationspartner wird jeweils das am stärksten korrelierende Feature des anderen Frames gewählt. Wenn zwei Features sich gegenseitig als bevorzugten Korrelationspartner haben werden diese verkettet. So entstehen über die Zeit Historien der Features (Abbildung 1).

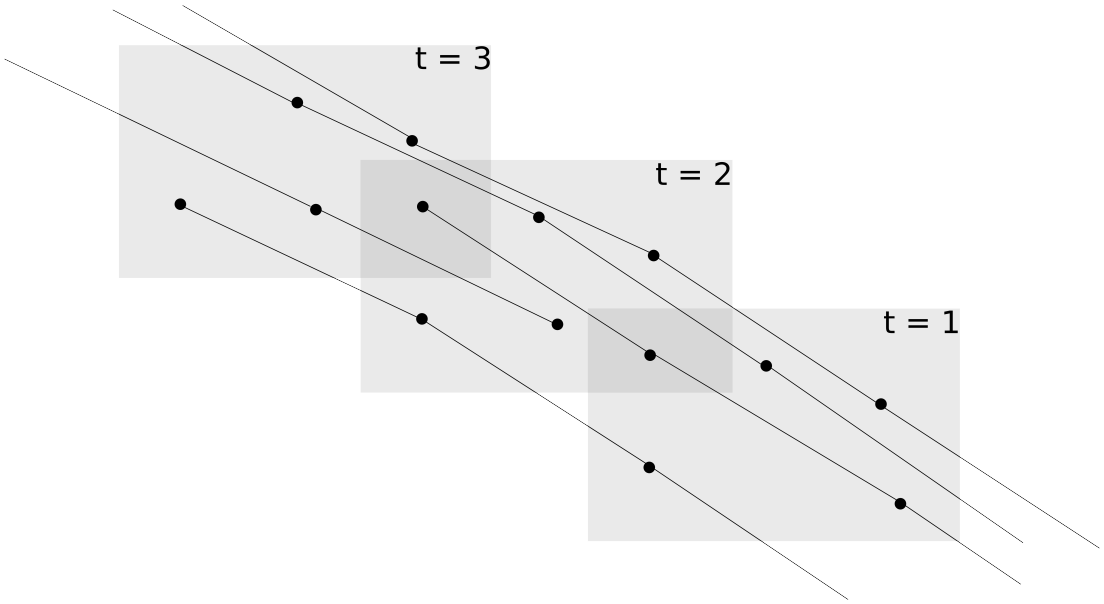

Abbildung 1: *Erkannte Features werden über mehrere Frames zu Historien verknüpft um die Bewegung der Kamera zu verfolgen.*

4 Bestimmung der Trajektorie

4.1 Überblick

Da nun zusammengehörige Punkte verkettet sind, lässt sich bei bekannten Raumkoordinaten dieser Punkte die Änderung der Kameraposition über die Zeit verfolgen. Die Bestimmung der Raumkoordinaten gestaltet sich in Abhängigkeit der Anzahl der Kameras unterschiedlich kompliziert. Im folgenden werde ich jeweils eine Variante mit einer einzelnen Kamera und mit einer Stereooptik vorstellen. Der Algorithmus läuft folgendermaßen ab:

1. Erstelle initiale 3D-Referenzpunkte.

2. Verfolge diese Punkte über mehrere Frames und bestimme jeweils die relative Position zu diesen.

3. Erstelle nach einigen Frames neue 3D-Referenzpunkte und fahre mit Schritt 2 fort.

4. Setze nach einigen Durchläufen eine Firewall.

4.2 Stereooptik

Bei dieser Variante werden zwei kalibrierte Kameras eingesetzt. Kalibriert heisst in diesem Zusammenhang, dass die relative Position der Kameras zueinander bekannt ist. Bei dieser Variante ist es naturgemäß möglich, vergleichsweise einfach zu jedem Zeitpunkt 3D Punkte aus den aufgenommenen Bildern durch Triangulation zu erstellen.

4.2.1 Initialisierung

Zur Bestimmung der späteren Bewegung werden inital Referenzpunkte in der Umgebung gesucht. Hierzu werden im linken und rechten Kamerabild jeweils Features gesucht (siehe Abschnitt 2). Anschliessend werden Korrelationen zwischen den Features beider Bilder hergestellt (siehe Abschnitt 3). Da die Positionen der Kameras zueinander und die zusammengehörigen Punkte in beiden Aufnahmen bekannt sind, können nun direkt 3D-Punkte trianguliert werden.

4.2.2 Punktverfolgung

Die Bewegung der bekannten Referenzpunkte wird über mehrere Frames verfolgt. Da diese Punkte in Wirklichkeit fest sind, entspricht diese Bewegung der Punkte im Bild einer Bewegung der Kamera und somit des Roboters. Hier spielen die Historien der Features eine Rolle, da diese ein Zuordnung über die Zeit herstellen. Da zu jedem Feature, das einen Referenzpunkt bildet, die Raumkoordinaten bekannt sind, lassen sich jeweils drei davon zu einem Dreieck mit bekannten Kantenlängen verbinden.

Aus der Lage dieses Dreiecks im Bild kann mit dem Dreipunktalgorithmus [1] die relative Lage dieser Punkte zu einer Kamera im Ursprung berechnet werden und somit auch die Position der Kamera in unserem Koordinatensystem. Hierzu sind mehrere Methoden bekannt, wie z.B. die Nullstellenbestimmung eines Polynoms vierten Grades, dass in geschlossener Form aufgestellt werden kann. Um ein gegen Ausreisser robustes Ergebnis zu erhalten, wird der Dreipunktalgorithmus als Hypothesengenerator in einem “preemptive RANSAC” genutzt und das Ergebnis noch iterativ verbessert.

4.2.3 Erstellung neuer Referenzpunkte

Nach einiger Zeit können die bekannten Referenzpunkte aus dem sichtbaren Bereich verschwinden. Daher müssen nach einigen Frames neue Features gesucht werden, die genau wie im ersten Schritt zu Raumpunkten trianguliert werden. Da die relative Position der Kamera durch die bisherige Trajektorie bekannt ist,

können die neuen Punkte sofort in das selbe Koordinatensystem gebracht werden und dienen als neue Referenzpunkte.

4.2.4 Firewall

Nach einigen Schritten wird eine "Firewall" eingesetzt. Diese verhält sich wie ein Neustart des Systems, das heisst die Liste der Referenzpunkte wird wieder geleert und erneut mit Schritt 1 des Algorithmus begonnen. Nur das Koordinatensystem wird durch die Firewall durchgereicht. Hiermit soll eine Fehlerakkumulation und -propagation durch die Verwendung relativer Positionen verhindert werden.

4.3 Monooptik

4.3.1 Überblick

Aus den Bildern einer Kamera können keine Tiefeninformationen gewonnen werden und somit nicht sofort 3D Referenzpunkte bestimmt werden. Um in diesem Fall die Position eines Punktes im Raum zu rekonstruieren, muss dieser aus verschiedenen Perspektiven betrachtet werden. Da sich die Kamera beim Einsatz zur Roboternavigation bewegt, stehen diese Perspektiven jedoch zur Verfügung. Das Grundprinzip ist zwar ähnlich wie bei der Stereooptik, der genaue Ablauf ändert sich aber zu:

1. Finde Features und verfolge diese über einige Frames.
2. Sobald mehrere Perspektiven dieser Features vorliegen, bestimme die Kamerapositionen mit dem 5-Punkt-Algorithmus und trianguliere die Punkte in 3D Referenzpunkte.
3. Verfolge diese Punkte über mehrere Frames und bestimme jeweils die relative Position zu diesen.
4. Trianguliere diese Punkte neu.
5. Setze eine Firewall ein.

Hierbei unterscheidet sich der Algorithmus nur in der Bestimmung der initialen Referenzpunkte signifikant vom oben beschriebenen Ablauf mit einer Stereooptik. Um hier bei unbekannten Kamerapositionen und unbekannten Raumpunkten zu Referenzpunkten zu kommen, ist es nötig, zuerst die Kamerapositionen zu bestimmen. Hierzu dient der Fünfpunktalgorithmus, den ich im nächsten Abschnitt vorstellen werde.

4.3.2 Five-Point Relative Pose Problem

Um aus den Aufnahmen einer Kamera Raumpunkte zu rekonstruieren, sind, wie oben erwähnt, mehrere Perspektiven nötig. Hierbei ergibt sich das Problem, dass einerseits diese Punkte benötigt werden um die Kamerapositionen zu den Aufnahmezeitpunkten zu bestimmen, anderseits die Kamerapositionen benötigt werden um die Raumpunkte zu triangulieren.

Es reicht an dieser Stelle die relativen Kamerakoordinaten zu bestimmen, da die erste Kameraposition bekannt ist und das Ergebnis daher später in das vorhandene Koordinatensystem gebracht werden kann. Somit ist für die erste Kameraposition die Annahme, sie befände sich im Ursprung und sei nicht rotiert ausreichend. Für die zweite Aufnahmeposition werden jetzt die Translation und Rotation der Kamera gesucht.

Da zur Bestimmung der Positionen nur Strahlrichtungen vom Projektionszentrum der Kamera zu den fünf Punkten zur Verfügung stehen, ist es unmöglich eine korrekte Skalierung zu erhalten, da bei Skalierung diese Richtungen gleich bleiben. Dadurch kann die Linie zwischen den beiden Projektionszentren als Linie fester Länge angenommen werden, wodurch die Translation zwischen den Aufnahmepositionen nur noch zwei Freiheitsgrade hat. Zusammen mit den drei Freiheitsgraden der Rotation ergeben sich also fünf Freiheitsgrade und somit werden zur Lösung des Problems mindestens fünf Punkte im Raum benötigt.

Hierzu sind einige Vorüberlegungen wichtig. Die Aufnahme eines Punktes $Q = [x, y, z, 1]$ entspricht einer perspektivischen Projektion auf die Bildebene der Kamera. Hierbei wird der Punkt erst mit der Matrix $[R|t]$ durch Rotation und Translation in das Koordinatensystem der Kamera gebracht und danach unter Berücksichtigung der Brennweite und anderer intrinsischer Kameraparameter (zusammengefasst in der Matrix K) auf die Bildebene projeziert. Die Gesamtabbildung eines Raumpunktes Q auf einen Bildpunkt q lässt sich also schreiben als

$$q \;=\; K[R|t]Q. \tag{10}$$

Die beiden Projektionszentren C_1 und C_2 der Kameras bestimmen zusammen mit dem Punkt Q die sogenannte epipolare Ebene (Die geometrischen Zusammenhänge sind in Abbildung 2 dargestellt). Die Abbildungen des Punktes auf die Bildebenen müssen somit ebenfalls auf dieser Ebene liegen. Durch den Schnitt der epipolaren Ebenen mit der Bildebene entsteht so eine Linie, auf der sich der Punkt in beiden Bildern befinden muss. Anders ausgedrückt bedeutet dies, dass die Vektoren t, der die Projektionszentren verbindet, $RK^{-1}q$ und $K^{-1}q'$, die die Projektionszentren mit dem Punkt Q verbinden, komplanar sind und das Spatprodukt $[a, b, c] = a\,(b \times c)$ verschwindet. Dieser Zusammenhang führt zur

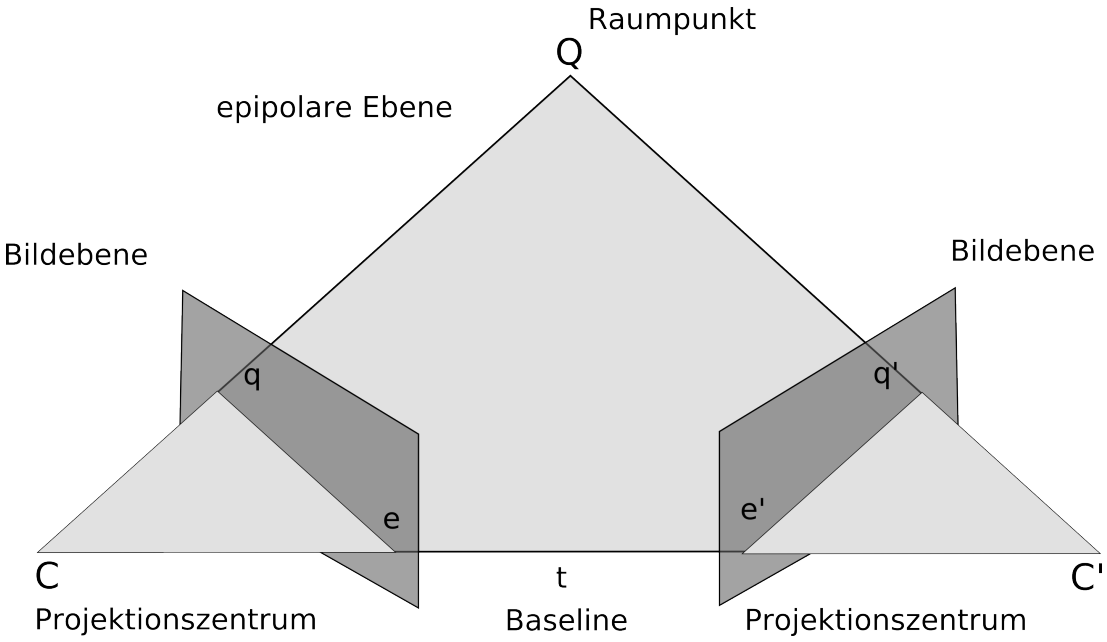

Abbildung 2: *Die epipolare Ebene wird durch die Projektionszentren C_1, C_2 und den Raumpunkt Q aufgespannt.*

Fundamentalmatrix F ($[t]_x$ bezeichnet eine Matrix mit $\forall v : [t]_x v = t \times v$):

$$F = K_2^T [t]_x R K_1^{-1} \tag{11}$$

Dies ist die Matrix, die für alle in den beiden Bildern korrespondierenden Punkte die Komplanaritätsbedingung erfüllt:

$$q'^T F q = 0 \tag{12}$$

Im folgenden wird davon ausgegangen, dass die intrinsischen Kameraparameter bekannt sind und die Vektoren q, q' mit diesen vormultipliziert wurden. Dadurch vereinfacht sich das Problem auf die Bestimmung der sogenannten Essentialmatrix

$$E = [t]_x R \tag{13}$$
$$q'^T E q = 0. \tag{14}$$

Weitere Bedingungen sind durch die folgenden beiden Theoreme gegeben:

Theorem 1: Eine relle nicht-triviale 3×3 Matrix F ist eine Fundamentalmatrix genau dann wenn gilt

$$\det(F) = 0 \tag{15}$$

und da eine Essentialmatrix zusätzliche Bedingungen an die Singulärwerte stellt gilt weiterhin folgendes Theorem:

Theorem 2: Eine reelle nicht triviale 3×3 Matrix E ist eine Essentialmatrix genau dann, wenn gilt

$$EE^T E - \frac{1}{2}\text{trace}\left(EE^T\right)E = 0. \tag{16}$$

Die Komplanaritätsbedingung lässt sich umschreiben in

$$\begin{aligned} \tilde{q}^T \tilde{E} &= 0 && (17)\\ \tilde{q} &= [q_1 q_1', q_2 q_1', q_3 q_1', q_1 q_2', q_2 q_2', q_3 q_2', q_1 q_3', q_2 q_3', q_3 q_3']^T && (18)\\ \tilde{E} &= [E_{11}, E_{12}, E_{13}, E_{21}, E_{22}, E_{23}, E_{31}, E_{32}, E_{33}]^T. && (19) \end{aligned}$$

Da die Matrix $\tilde{E}$ die Komplanaritätsbedingung für alle Punkte erfüllt, gilt auch

$$\tilde{q}_1^T \tilde{E} + \tilde{q}_2^T \tilde{E} + \tilde{q}_3^T \tilde{E} + \tilde{q}_4^T \tilde{E} + \tilde{q}_5^T \tilde{E} = 0 \tag{20}$$

Jetzt lässt sich eine Matrix A mit den Spalten $\tilde{q}_i$ $i \in 1, \ldots, 5$ und $A^T \tilde{E} = 0$ aufstellen. Nun sucht man nach vier Vektoren, die den Nullraum der Matrix A aufspannen um aus diesen die Matrix E derart zu konstruieren, dass diese die obige Bedingung erfüllt. Diese Vektoren lassen sich direkt aus einer Singulärwertzerlegung ablesen, jedoch wird hier eine effizientere Methode mit einer angepassten QR-Zerlegung verwendet (siehe [4]). Die gefundenen Vektoren korrespondieren direkt zu vier Matrizen X, Y, Z, W mit denen sich die Essentialmatrix konstruieren lässt:

$$E = xX + yY + zZ + wW \tag{21}$$

Durch einsetzen von (21) in (16) lässt sich nun ein Gleichungssystem aufstellen, mit dem sich Koeffizienten x, y, z, die die Bedingungen (14) und (16) erfüllen, bestimmen lassen, wobei $w = 1$ angenommen wird. Nach erfolgter Gauss-Jordan-Elimination mit Pivotisierung bekommt man folgendes Gleichungssystem (mit Skalaren . und Polyonomen N-ten Grades in der Variable z [N]):

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
a	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
b		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
c			1	.	.	.	.	.	.	.	[2]	[2]	[3]
d				1	.	.	.	.	.	.	[2]	[2]	[3]
e					1						[2]	[2]	[3]
f						1					[2]	[2]	[3]
g							1				[2]	[2]	[3]
h								1			[2]	[2]	[3]
i									1		[2]	[2]	[3]
j										1	[2]	[2]	[3]

Das Gleichungssystem bis hierhin zu berechnen reicht, da die restlichen zur Bestimmung der Koeffizienten nötigen Zeilen aus Effizienzgründen explizit berechnet werden werden:

$$k = \langle e \rangle - z * \langle f \rangle \quad (22)$$
$$l = \langle h \rangle - z * \langle g \rangle \quad (23)$$
$$m = \langle j \rangle - z * \langle i \rangle \quad (24)$$

Diese Gleichungen lassen sich wieder in eine Matrix schreiben. Diese Matrix bildet $[x, y, 1]$ auf null ab, und damit muss die Determinante verschwinden. Die Determinante ist ein Polynom vom Grad 10, dessen Nullstellen Lösungen für die Koeffizienten der Essentialmatrix ergeben. Als effizienten Weg hierzu schlagen die Autoren Sturm-Sequenzen vor. Dies sind 10 korrespondierende Polynome niedrigeren Grades (siehe [4]). Bei diesem Verfahren werden Intervalle eingegrenzt in denen sich die Nullstellen befinden und somit erreicht man in einem Folgeschritt eine schnelle Konvergenz von Verfahren zur iterativen Nullstellenbestimmung (wie z.B. dem Newtonverfahren). Sind die Nullstellen gefunden, können x und y sofort errechnet werden. Da ein Polynom 10. Grades jedoch bis zu 10 reelle Nullstellen haben kann, gibt es mehrere Lösungen für E.

Für jede Lösung müssen nun die Rotation und Translation bestimmt werden. Zur Rekonstruktion der Rotationsmatrix und des Translationsvektors aus der Essentialmatrix dient folgendes Theorem (siehe [3]):

Theorem 3: Sei die Singulärwertzerlegung der Essentialmatrix $E = U\mathrm{diag}\,(1, 1, 0)\, V^T$, mit $\det(U) > 0$ und $\det(V) > 0$. Dann ist $t = [u_{13}, u_{23}, u_{33}]$ und entweder $R = UDV^T$ oder $R = UD^TV^T$.

$$D = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (25)$$

Hier sind vier Lösungen möglich, die korrekte Lage, die gespiegelte hierzu sowie jeweils eine rotierte. Nur bei der korrekten Lösung liegen alle Bildpunkte vor den beiden Bildebenen. Es muss also lediglich auf die Erfüllung dieser Bedingung getestet werden und gegebenenfalls nacheinander die drei möglichen Transformationen angewandt werden um zur gesuchten Lösung zu kommen.

Um nun eine eindeutige Lösung zu bekommen wird in einem dritten Bild die Kameraposition an Hand der bekannten fünf Punkte mit dem Dreipunktalgorithmus bestimmt. Als Hypothese wird dem RANSAC die Lösung zurückgegeben, die in den drei Perspektiven nach Triangulation der Punkte den geringsten Fehler aufweist.

4.3.3 Bestimmung der Trajektorie

Da jetzt relative Positionen der Kamera über die Historie des Features bekannt sind, können die Features in Referenzpunkte trianguliert werden. Anhand dieser Punkte wird jetzt die Trajektorie der Kamera in den für den Fünfpunktalgorithmus genutzten Frames bestimmt. Um das Ergebnis nun in das bestehende Koordinatensystem einzubetten muss es noch passend skaliert, rotiert und verschoben werden. Für diesen Schritt wird wieder ein "preemptive RANSAC" genutzt.

Wie beim Fall mit einem Stereoaufbau werden auch in diesem Fall die Positionen der Kamera mit dem einfacheren Dreipunktalgorithmus bestimmt. Gelegentlich werden die bekannten Referenzpunkte neu trianguliert unter Zuhilfename der bis zu diesem Zeitpunkt ersten und letzten Beobachtung des entsprechenden Punktes – und wieder beim Schritt 1 gestartet. Nach einigen derartigen Durchläufen wird auch, wie im Stereofall, eine Firewall eingesetzt.

5 Zusammenfassung

Das in dieser Seminararbeit vorgestellte Verfahren zur Echtzeitpositionsbestimmung wurde erfolgreich zur Roboternavigation sowohl am Boden als auch in der Luft getestet. Als Ground Truth waren die Roboter mit Differential GPS ausgestattet. Nach mehreren hundert Metern zurückgelegter Strecke in unebenen Gelände sowie mit eng gefahrenen Kreisen betrug die Differenz in der gemessenen zurückgelegten Entfernung zwischen DGPS und Visual Odometry weniger als 5% der Strecke. Auch die Trajektorien weisen nur eine geringe Differenz auf. Damit ist Visual Odometry ein Verfahren mit einer ähnlichen Präzision wie andere sensorbasierte Ansätze und eignet sich als Grundlage für SLAM Verfahren und zur Navigation.

Literatur

[1] R. M. Haralick, K. Ottenberg, and M. Nolle. Analysis and solutions of the three point persepective pose estimation problem. In *Proceedings Computer Vision and Pattern Recognition '91*, pages 592–598, Lahaina, Maui, June 1991.

[2] Chris Harris and Mike Stephens. A combined corner and edge detector. *?*, pages 147–151, 1988.

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[4] David Nistér. An efficient solution to the five-point relative pose problem. In *CVPR*, pages 195–202. IEEE Computer Society, 2003.

[5] David Nistér, Oleg Naroditsky, and James R. Bergen. Visual odometry. In *CVPR (1)*, pages 652–659, 2004.