Farbsegmentierung für Aktives Sehen

J. Denzler, B. Heigl, D. Paulus

Lehrstuhl für Mustererkennung (Informatik 5)
Universität Erlangen–Nürnberg
Martensstr. 3
91058 Erlangen (Germany)
{denzler,paulus}@informatik.uni-erlangen.de
http://www5.informatik.uni-erlangen.de

1 Einleitung

Eines der Ziele des Aktiven Sehens ist die Konturverfolgung bewegter Objekte. In unserer Anwendung soll in einer Szene ein bewegtes Objekt mit Roboter und Kamera verfolgt werden. Die bisherigen in der Literatur zu findenden Ansätze beinhalten meist Methoden, die ausschließlich auf Grauwertbildern basieren. In diesem Artikel werden Verfahren untersucht, die durch Verwendung von Farbinformation die Detektion des bewegten Objektes stabilisieren sollen. Hierzu werden zwei Segmentierungsalgorithmen untersucht, die beide einer Methode von Dubuisson und Jain in [2] entnommen sind.

Zur Verbesserung dieser Ergebnisse werden im folgenden Farbdifferenzen in anderen Farbräumen untersucht. Außerdem ermöglichen algorithmische Verbesserungen kürzere Laufzeiten und robustere Schwellwerte (Abs. 2). Schließlich wird dargestellt, inwiefern diese verbesserte Methode für das Aktive Sehen geeignet ist (Abs. 3), bzw. welche Modifikationen notwendig sind. Durch Verwendung unterschiedlicher Auflösungen und Fovealen Sehens wird die Laufzeit der Algorithmen reduziert, damit bei Integration in einen Sensor–Aktor–Zyklus die Objektverfolgung in Echtzeit durchgeführt werden kann (Abs. 4). Weitere Arbeiten in diesem Anwendungszusammenhang werden kurz erläutert (Abs. 5).

2 Farbregionen-Segmentierer

Im folgenden soll die Methode des Farbregionen–Segmentierers vorgestellt werden, wie sie in [2] vorgeschlagen wurde und welche Verbesserungen hinzugefügt wurden. Der Segmentierer basiert auf einem "Split–and–Merge" Algorithmus [4], der Farbdifferenzen verwendet. Zu Beginn wird das Bild in einem Quadtree organisiert. Hierzu wird vorausgesetzt, daß das Bild quadratisch, und seine Kantenlänge eine Zweierpotenz ist. Es erfolgt eine initiale Aufteilung in Teilquadrate, was einem Abstieg im Quadtree entspricht. Wie weit abgestiegen wird, muß durch einen Parameter entsprechend der gewünschten Genauigkeit (s.u.) gewählt werden.

Beim Split–Vorgang wird ein inhomogenes Quadrat in vier Unterquadrate aufgeteilt, zu deren Bestimmung der Farb–Mittelwert und die Farb–Varianz berechnet werden. Der Farb–Mittelwert $\boldsymbol{C}_{Q_{x,y}}$ über das Quadrat $Q_{x,y} = \{(\mu,\nu)|\mu \in [x, x+n-1], \nu \in [y, y+n-1]\}$ mit der Kantenlänge n ergibt sich aus den Farbwerten $\boldsymbol{f}(x,y) = (f_r(x,y), f_g(x,y), f_b(x,y))^T$ an den Positionen (x,y) innerhalb von $Q_{x,y}$ durch Mitteln der einzelnen Farbkanäle:

$$\boldsymbol{C}_{Q_{x,y}} = \frac{1}{n^2} \sum_{(l,m)\in Q_{x,y}} \boldsymbol{f}(l,m)$$

Die Varianz wird in [2] als Summe der Varianzen der einzelnen Farbkanäle berechnet. Um sich vom RGB–Farbraum zu lösen und beliebige Farbabstandsmaße zu verwenden, wurde die

ursprüngliche Definition der Farb–Varianz $Var(Q_{x,y})$ im Quadrat $Q_{x,y}$ folgendermaßen modifiziert, wobei $D(\boldsymbol{C}_1, \boldsymbol{C}_2)$ den Farbabstand zwischen den Farbvektoren $\boldsymbol{C}_1$ und $\boldsymbol{C}_2$ bezeichnet:

$$Var(Q_{x,y}) = \frac{1}{n^2} \sum_{(l,m) \in Q_{x,y}} \left[D(\boldsymbol{f}(l,m), \boldsymbol{C}_{Q_{x,y}}) \right]^2$$

Gute Ergebnisse wurden hier mit dem uniformen Farbraum CIE–(L*u*v*) erzielt [3], wobei für D hier der euklidische Abstand gewählt wurde.

Ist ein Quadrat $Q_{x,y}$ nun genügend inhomogen, d.h. überschreitet die Varianz $Var(Q_{x,y})$ eine gegebene Schwelle t_v, so wird das Quadrat in vier Unterquadrate aufgeteilt ("split"). eurch diesen Vorgang wird der Quadtree an gewissen Stellen immer tiefer, das Bild wird immer feiner aufgegliedert. Man bestimmt eine minimale Größe für die Quadrate, bei der die Aufteilung nicht mehr weiter fortgesetzt wird. So kann die Auflösung des Ergebnisses variiert werden, was sich auch auf die Rechenzeit auswirkt. Hierauf wird später noch unter dem Blickwinkel Aktives Sehen (Abs. 3) eingegangen.

Beim Vereinigungs–Vorgang ("merge") wird, falls der Farbabstand unter vier Quadraten, die im Quadtree einen gemeinsamen Elternknoten besitzen, unterhalb einer Schwelle t_m liegt, der Baum an dieser Stelle verkürzt; die vier Unterquadrate werden also miteinander verschmolzen. Auch hier ist der ursrüngliche RGB–Farbabstand ersetzt worden durch Farbabstandsfunktionen D in beliebiegen Farbräumen.

Anschließend werden die so ermittelten Quadrate zu Regionen gruppiert, falls sie farblich ähnlich sind. In [2] werden zwei Quadrate zu einer Region zusammengefügt, falls der RGB–Farbabstand unter einer gewissen Schwelle t_g liegt. Es werden alle Nachbarn, die diesem Kriterium genügen, zur Region hinzugefügt. Beim Zusammenfügen wird der gesamte Farb–Mittelwert aus den beiden Teilregionen durch Gewichtung mit der jeweiligen Größe ermittelt. Anschließend wird die nächste Region erzeugt. Das Problem bei dieser Vorgehensweise ist, daß es möglich ist, daß ein Quadrat nicht zu der farblich nächstliegenden Region hinzugenommen wird, sondern einfach zu derjenigen, die als erstes versucht, dieses Quadrat an sich zu binden. Dadurch blähen sich manche Regionen ungewollt auf und andere kommen wiederum zu kurz. Außerdem ist die Wahl der Schwellwerte erschwert.

Um dies zu beseitigen wurde folgendermaßen vorgegangen. Zu Beginn wird jedes Quadrat des Quadtrees als Region interpretiert. Anschließend wird jeweils an eine Region die farblich nächstliegende Nachbarregion d.h. diejenige Region mit dem geringsten Farbabstand hinzugefügt, sofern dieser unter einer gewissen Schwelle liegt. Auf diese Weise werden alle Regionen nacheinander immer wieder durchgegangen, bis keine Regionen mehr verbunden werden können. Diese Methode liefert wesentlich bessere Ergebnisse als die vorher genannte, denn die Regionen vergrößern sich gleichzeitig in allen Bereichen des Bildes. Dadurch ergibt sich jedoch häufig die Situation, daß in gewissen Bereichen des Bildes keine Änderungen mehr stattfinden, aber für jede Region bei jedem Durchlauf überprüft wird, ob eine Nachbarregion hinzugefügt werden kann. Um dies zu verhindern, wird der Fall, daß keiner der Nachbarn hinzugefügt werden kann, vermerkt und diese Region beim nächsten Durchlauf nicht mehr betrachtet. Dieser Vermerk wird wieder aufgehoben, falls eine Nachbarregion sich mit einer anderen verbinden konnte. Die Wahl des Schwellwertes wird durch dieses Vorgehen robust und die so erhaltenen Regionen entsprechen den homogenen Bereichen im Bild qualitativ gut. Außerdem werden zur Berechnung des Farbabstands wiederum beliebige Farbabstandsmaße in Betracht gezogen. Die Verbesserungen des Verfahrens wurden hauptsächlich im Gruppierungsschritt vorgenommen, der den letzten Schritt der Farbregionen–Segmentierung darstellt, deren Ablauf in Bild 1 zusammengefaßt ist.

3 Aktives Sehen

Zur Laufzeitverkürzung für die Integration in einen Sensor-Aktor-Zyklus sind Methoden aus dem Aktiven Sehen anwendbar. Beispielsweise kann — wie schon erwähnt — die Rechenzeit durch Änderung der Auflösung variiert werden, indem man die minimale Größe der Quadrate angibt, ab der nicht mehr unterteilt werden darf. Die Ergebniskonturen besitzen dann entsprechend eine gröbere oder feinere Auflösung. Ein anderer Ansatz besteht darin, daß man voraussetzt, daß sich das bewegte Objekt bzw. die interessanten Bereiche eher in der Mitte des Bildes als am Rand

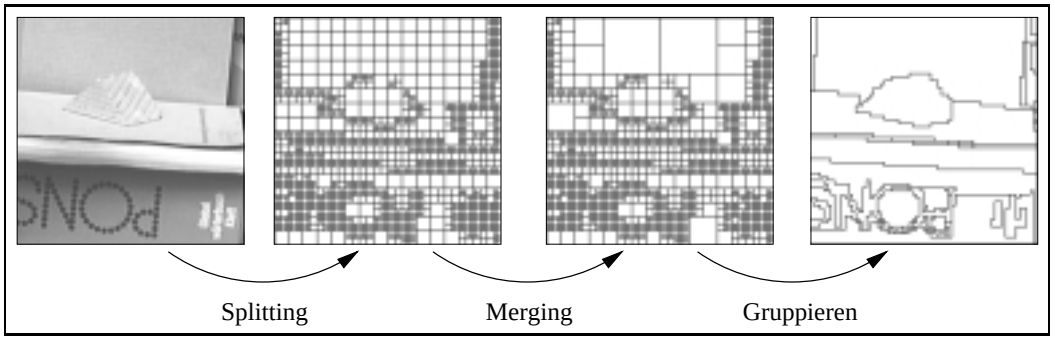

Bild 1: Beispielablauf des Regionensegmentierers. Das farbige Prisma weist im Grauwertbild nur einen ungenügenden Kontrast auf und wäre dort schwer zu detektieren.

befinden. Somit wird die Rechenzeit reduziert, indem man die Auflösung im mittleren Bereich des Bildes höher als am Rand gestaltet. Diese Überlegung ergibt sich aus der Nachahmung des menschlichen Auges, das ebenfalls die Eigenschaft besizt, in der sogenannten Fovea höher aufzulösen als im Randbereich. Deshalb wird diese Methode auch als Foveales Sehen bezeichnet. Hier wird sie dadurch verwirklicht, daß man die maximale Tiefe des Quadtrees in den verschiedenen Bildbereichen unterschiedlich festsetzt. Durch die Konzentration auf die Bildmitte bzw. durch die Vernachlässigung der Randbereiche wird die Rechenzeit reduziert (Abs. 5).

4 Anwendung

Der oben beschriebene Farbregionen-Segmentierer soll nun in einem Echtzeit–Konturverfolgungssystem Verwendung finden. Dubuisson und Jain gehen in [2] dabei so vor, daß sie ein Farb–Differenzbild–Verfahren anwenden. Dabei werden drei Frames $\boldsymbol{f}^{(1)}$, $\boldsymbol{f}^{(2)}$ und $\boldsymbol{f}^{(3)}$ mit einer statischen Kamera aufgenommen. Nach Anwendung eines Tiefpaßfilters, der die Änderungen durch Rauschen minimiert, werden folgende Differenzen berechnet:

$$\begin{aligned} d_r(x,y) &= |f_r^{(1)}(x,y) - f_r^{(2)}(x,y)| \cdot |f_r^{(2)}(x,y) - f_r^{(3)}(x,y)| \\ d_g(x,y) &= |f_g^{(1)}(x,y) - f_g^{(2)}(x,y)| \cdot |f_g^{(2)}(x,y) - f_g^{(3)}(x,y)| \\ d_b(x,y) &= |f_b^{(1)}(x,y) - f_b^{(2)}(x,y)| \cdot |f_b^{(2)}(x,y) - f_b^{(3)}(x,y)| \end{aligned}$$

Als Gesamtdifferenz wird die maximale Differenz der einzelnen Kanäle verwendet:

$$d(x,y) = \max\{d_r(x,y), d_g(x,y), d_b(x,y)\}$$

Alle Punkte deren Wert $d(x,y)$ über einer zu bestimmenden Schwelle liegt, befinden sich auf bewegten Kanten. Auf den Kanten in Bewegungsrichtung tauchen allerdings keine Punkte auf, weil sich in der Bildinformation keine Änderung feststellen läßt.

Als Verbesserung der obigen Differenzberechnung wurde folgende Formel eingeführt, um beliebige Farbabstandsmaße D verwenden zu können:

$$d(x,y) = D(\boldsymbol{f}^{(1)}(x,y), \boldsymbol{f}^{(2)}(x,y)) \cdot D(\boldsymbol{f}^{(2)}(x,y), \boldsymbol{f}^{(3)}(x,y))$$

Wie man in Bild 2 gut erkennen kann, erhält man durch Bilden der konvexen Hülle um die so ermittelten Punkte eine Ergebniskontur, die als Vorhersage für die Kontur des bewegten Objektes verwendet werden kann. Sie ist im allgemeinen jedoch zu groß. Deshalb wird dieses Ergebnis der Bewegungs–Segmentierung mit dem der Regionen–Segmentierung verknüpft, indem man die

Bewegungs–Kontur als Schablone betrachtet, innerhalb der sich alle Regionen befinden, die Teil des bewegten Objektes sind. Die Ergebniskontur ergibt sich als Umrandung all derjenigen Farbregionen, deren prozentualer Anteil der Pixel, die sich innerhalb dieser Schablone befinden, über einer festzusetzenden Schwelle liegt.

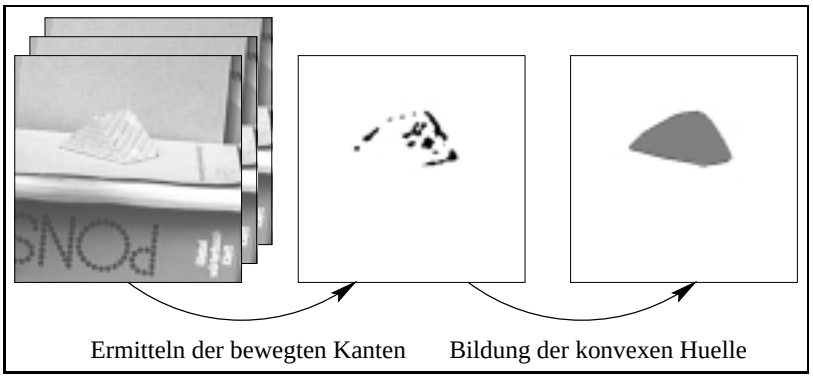

Bild 2: Beispielablauf des Bewegungssegmentierers.

Bei dieser Vorgehensweise kann die Rechenzeit dadurch verringert werden, daß man zuerst den Bereich des Bildes ermittelt, in dem sich bewegte Kanten befinden. Anschließend legt man um diesen ein Rechteck mit einer Zweierpotenz als Kantenlänge und segmentiert innerhalb dieses Rechtecks die Farbregionen. Erst dann wendet man die obige Kombination der Ergebnisse an.

Die Implementierung erfolgte in C++ unter Unix auf HP Workstations.

In unseren Experimenten zeigte sich das Verfahren robust gegenüber der Wahl der Schwellwerte; in den Versuchen verwendeten wir für t_v und t_m jeweils den Wert 5 und einen Wert von 10 für t_g. Mit Bildern der Größe 256×256 wurden auf einer HP 735/99 Laufzeiten von 4.0 Sekunden für die Farbregionen–Segmentierung ermittelt. Damit ergab sich eine Reduktion von ca. 50 % gegenüber den ersten Versuchen mit dem Verfahren aus [2] unter vergleichbaren Bedingungen.

5 Ausblick

Die Laufzeitverbesserungen durch die Methoden des Aktiven Sehens werden derzeit untersucht. Mit den hier und in [3] dargestellten Verfahren sind Verbesserungen und Erweiterungen des Systems vorgesehen, das in [1] beschrieben ist. Damit wird eine Verkürzung der in Abs. 4 angegebenen Laufzeiten erreicht.

Literatur

[1] J. Denzler and D. Paulus. Active motion detection and object tracking. In *Proceedings of the International Conference on Image Processing (ICIP)*, volume 3, pages 635–639, Austin, TX, USA, November 1994. IEEE Computer Society Press.

[2] M. Dubuisson and A.K. Jain. Object contour extracting using color and motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 471–476. Ney York City, 1993.

[3] B. Heigl. Arbeitstitel: Untersuchung von Farbräumen zur Merkmalsgewinnung in Bildfolgen. Studienarbeit, IMMD 5 (Mustererkennung), Universität Erlangen–Nürnberg, Erlangen, erscheint 1995.

[4] S.L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *J. Assoc. Comput. Mach.*, 23:368–388, 1976.