Bildentrauschung mit Variationsansätzen.

Totale Variation Minimierung

Die Minimierung der Totalvariation basiert auf Level-Set-Näherungen. Eines unserer Hauptwerkzeuge ist der sogenannte Fast-Level-Set-Transform (FLST). Dieser Algorithmus stellt das Bild auf geeignete Weise dar, speichert das Ergebnis in einem Baum, und liefert uns eine vollständige Repräsentation des Bildes.
Daraus gewinnen wir einen Algorithmus zur Minimierung derTotalvariation der auf Level-Sets basiert.

1.) Einleitung:

Die Information in Bildern kann aufgrund von Aufnahme- oder Übermittlungsfehlern gestört sein. Diese Störungen nennt man Rauschen. Dadurch lässt sich das Bild nur schlecht weiterverwenden, deshalb benötigen wir sogenannte Filter.
Es ist bekannt, dass Bildentrauschungsmethoden, welche auf Glättung beruhen, zwar auf der einen Seite das Rauschen beheben, aber auf der anderen Seite auch wichtige Details des Bildes, wie Ecken und Kanten.
Ausserdem sind Bilder, die als Funktionen in 2 Variablen beschrieben werden, höchstens stückweise stetig, und wichtige Details, wie eben die Ecken sind Unstetigkeitsstellen.
Deshalb erlauben wir Unstetigkeiten im Ergebnis der Minimierung und können somit scharfe Kanten erhalten.
Im folgenden bezeichnen wir die Bildfunktion mit u, wobei u ist eine reellwertige Funktion ist, die auf der offenen Teilmenge $\Omega \subset \mathbb{R}^2$ definiert ist (wobei Ω normalerweise ein Rechteck ist.).
Dann besteht Rudin und Osher`s Prinzip einfach aus dem folgenden Problem:

Minimiere $\int_\Omega |\nabla u| = \int_\Omega \sqrt{(\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2}\, d(x, y)$ (1.1)

Nun kann die Minimierung der Totalen Variation erreicht werden durch einen 'Gradientenabstieg' welcher zu folgender Evolutionsgleichung führt:

$$\frac{\partial u}{\partial t} = div(\frac{\nabla u}{|\nabla u|}) = curv(u). \qquad (1.2)$$

Das Hauptproblem bei der oben genannten Gleichung liegt im Integraten $|\nabla u|$, da er unter Umständen nicht überall definiert ist.

2.) Ein Praktischer Algorithmus:

Normalerweise sind Bilddaten auf einem diskreten Gitter definiert und die Pixel können nur eine endliche Menge an Werten annehmen, die sogenannten Grauwerte.
Um das ganze auf geeignete Weise darstellen zu können, beschreiben wir zunächst unser Hauptwerkzeug für die Repräsentation von Bildern, den sogenannten FLST vor.

Fast Level Set Transform (FLST)

Für ein Bild u bezeichnen wir $X^\lambda=\{x|u(x)\geq\lambda\}$ und $X_\mu=\{x|u(x)\leq\mu\}$ als den oberen Levelset von λ und den unteren Levelset von μ.
Die Daten $(X^\lambda,\lambda\in\mathbb{R})$ oder $(X_\mu,\mu\in\mathbb{R})$ reichen aus, um das Bild u zu rekonstruieren als $u(x)=sup\{\lambda|x\in X^\lambda\}=inf\{\mu|x\in X_\mu\}$.

Weil eine zusammenhängende Komponente eines Levelsets auch Löcher haben kann, und wir aber Flächen ohne Löcher betrachten wollen, werden sogenannte Shapes eingeführt. Ein Shape ist eine zusammenhängende Komponente eines Levelsets bei dem die Löcher aufgefüllt wurden.
Der FLST zerlegt ein Bild in seine Shapes und konstruiert eine Baumstruktur um die Inklusionen der jeweiligen Shapes zu beschreiben, das führt zu einer vollständigen Repräsentation der Bildes.
Ein FLST – Modul hat als Output:
- Die Familie der Shapes, angeordnet in einer Baumstruktur (Wenn $C\subset F$, dann heisst F Elternshape von C)
- Für jeden Shape wird seine Fläche, sein Umfang, sein Grauwert λ und ob er einem oberen Levelset oder unteren Levelset angehört, gespeichert.

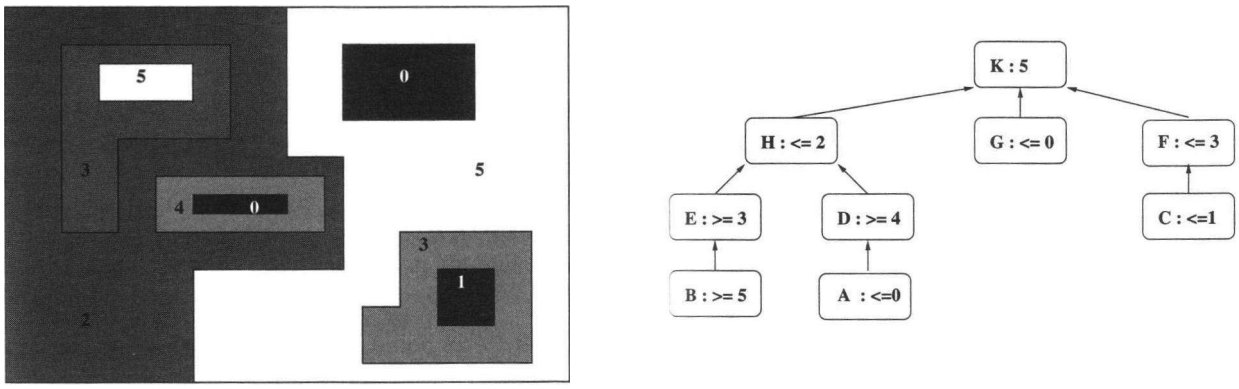

Abbildung 1

In Abbildung 1 sehen wir ein Graustufenbild, und den Baum, den der Algorithmus geliefert hat.

Nun wollen wir den Algorithmus vorstellen, für den Fall, dass das Bild u als bilineare Interpolation seiner Daten dargestellt wurde. In diesem Fall ist das Bild als eine stetige Funktion modelliert, und die Höhenlinien sind definiert als die zusammenhängenden Komponenten von Isolevelsets. Dieses Modell ist gut geeigent für unsere Ziel:
Die Höhenlinien sind glatter als wenn man annehmen würde, dass das Bild auf jedem Pixel konstant ist und weniger empfindlich auf Diskretisierung. Das führt zu einer besseren Abschätzung der Länge der Höhenlinie. Das Problem bei einer stetigen Funktion ist, dass sie eine unendlich Anzahl an Höhenlinien besitzt, und wir deshalb keine Shapes bilden könnten. Deshalb betrachten wir nur

die Punkte auf einem vorher bestimmten Gitter und die Sattelpunkte die die Funktion besitzt.

Auf einer Fläche, welche durch 4 in dem ursprünglichen Gitter benachbarte Punkte begrenzt ist, (ein sogenanntes 'Qpixel') sind die Höhenlinien hyperbolisch und der Sattelpunkt ist der gemeinsame Symmetriepunkt, sofern in dieser Fläche einer enthalten ist.

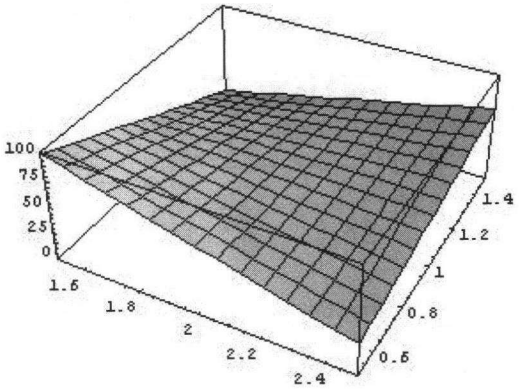

Abbildung 3

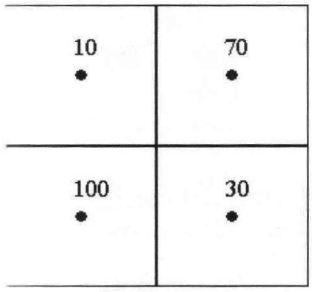

Abbildung 2

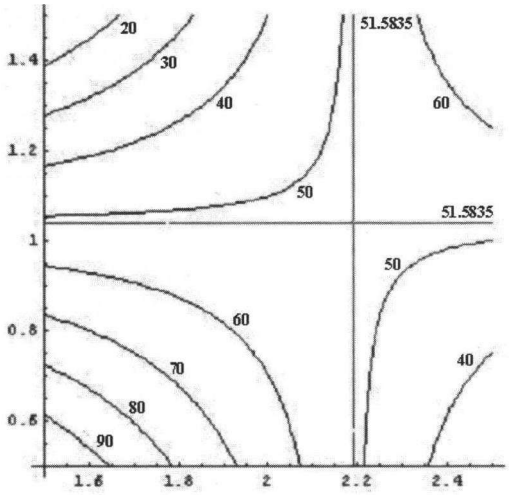

Abbildung 4

Abbildung 2 zeigt einen Qpixel, und Abbildung 3 die zugehörige, stetige Funtktion. In Abbildung 4 kann man die Höhenlinien und den Sattelpunkt betrachten.

Die Ausgabe des Algorithmus wird Tree of Bilinear Level Lines (TBLL) genannt.
Der Algorithmus arbeitet in zwei Schritten: Zuerst extrahiert er den anfänglichen TBLL, welcher zu den Höhenlinien gehört, die durch die Punkte oder durch die Sattelpunkte des ursprünglichen Gitters gehen. Aus diesem ursprünglichen TBLL kann später der TBLL für eine beliebige Anzahl von Leveln extrahiert werden.
Zuerst müssen wir diejenigen Qpixel finden, die Sattelpunkte enthalten, und den zugehörigen Funktionswert des Sattelpunktes berechnen. Ein Qpixel enthält genau dann einen Sattelpunkt, wenn das Maximum von 2 Werten die sich diagonal gegenüberliegen, kleiner ist, als das Minimum der anderen beiden Werte. Im beschriebenen Algorithmus ist ein Punkt entweder ein Mittelpunkt eines Pixels oder ein Sattelpunkt. Jeder hat einen zugeordneten Wert, d.h wir speichern die Werte der Pixelmittelpunkte und die Werte der Sattelpunkte.

Wir ordnen jedem Punkt P einen Shape S_P zu, wobei S_P der kleinste Shape ist, der P enthält. Initialisiert ist S_P mit NULL.
Wir durchsuchen alle Pixelpunkte P, und arbeiten die folgenden Schritte ab, wobei λ erstmal der Grauwert des jeweiligen Punktes P ist:

1.) Initialisiere eine Menge P, die Punkte enthalten soll, mit $\emptyset$, und eine Menge N die Nachbarpunkte entahlten soll mit {P}.
2.) Solange $N_\lambda \neq \emptyset$, wobei N_λ die Menge der Punkte aus N zum Level λ ist, entnehme N_λ von N und füge es zu zu P hinzu, und gib zu N die Nachbarn von N_λ hinzu, welche nicht schon in P enthalten sind.
3.) Wenn die Menge der Punkte in P kein Loch haben, und die Punkte von N alle vom Level $< \lambda$ oder alle vom Level $> \lambda$ sind, wird ein neuer Shape gebildet, der zu den Punkten aus P gehört. Andernfalls werden alle Punkte des Bildes die in P gespeichert sind auf den Level λ gesetzt und beenden.
4.) Jeder Punkt $Q \in P$, sofern S_Q = NULL ist, kommt in den neuen Shape. Andernfalls wandert man den Baum nach oben, angefangen bei S_Q und bildet den nun resultierenden Shape als Kind des neuen Shapes.
5.) Setze λ auf den nächsten Level der Punkte in N und gehe zurück zu Schritt 2.).

Nun wollen wir festlegen, was wir mit Nachbar meinen: Die Nachbarn eines Pixelmittelpunktes P sind die 4 Nachbarn von P und die Sattelpunkte in den Qpixeln in denen P eine Ecke ist. Die Nachbarn eines Sattelpunktes Q sind die Pixelmittelpunkte, die die Ecken des zugehörigen Qpixels bilden.
Nachdem das Bild bearbeitet wurde, sind seine Grauwerte vertauscht worden und es wird einfarbig. Das daraus entstehende, konstante Niveau ist das der Baumeswurzel.
Aus dem Wissen über den anfänglichen TBLL und einer gegebenen Diskretisierung kann man direkt den Ergebnis-TBLL berechnen. Für jeden Shape S im Original TBLL muss man das Level der Diskretisierung, welches zwischen den Leveln von S und den Leveln seiner Eltern S' im Original TBLL eingeschlossen ist, finden. Nun muss man einen Shape für jedes einzelne Level bilden.

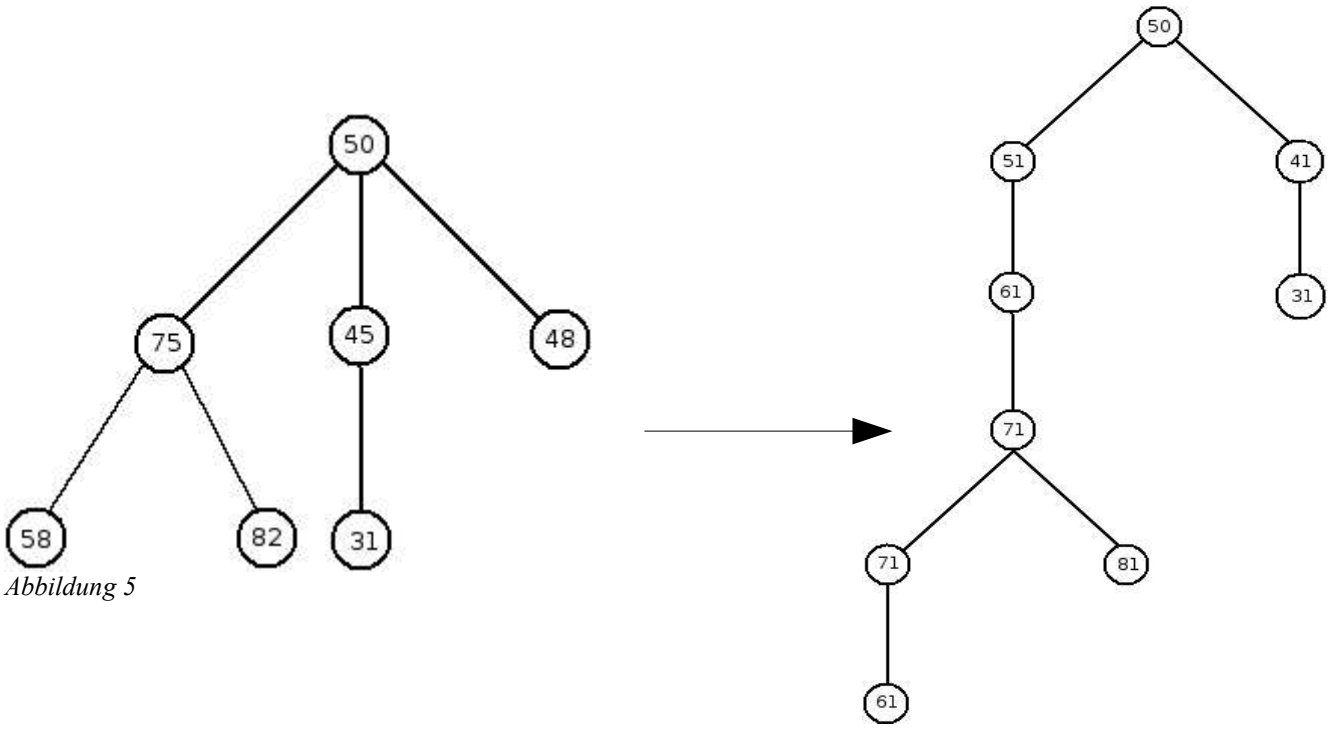

Abbildung 5

Links sieht man den anfänglichen TBLL, den der Algorithmus geliefert hat, und rechts den Baum der zum selben Bild gehört, wobei wir die Niveaumengen {1, 11, 21... } betrachten

3.) Der TV-Algorithmus

Wir werden (1.2) umformen, indem wir die Coarea Formel für Levelsets von u benutzen.
Das führt dazu, dass wir für fast alle $\lambda \in \mathbb{R}$ die folgende Gleichung betrachten können:

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = P(E_\lambda) \qquad (3.1)$$

wobei $E_\lambda = \{x | u(x) < \lambda\}$ das Levelset assoziiert zu λ und $P(E_\lambda)$ sein Umfang ist.
Zuerst betrachten wir ein $u \in BV(\Omega) \cap C^\infty(\Omega)$ und nehmen an, dass $\frac{\partial u}{\partial t}$ konstant ist, auf jeder Höhenlinie ∂E_λ .

Eine Globale Näherung für TV – Minimierung.

Eigenschaften von BV-Funktionen (Funktionen mit beschränkter Variation):

Sei $\Omega \subset \mathbb{R}^2$ offen, $C_c^1(\Omega, \mathbb{R}^2)$ sei die Menge aller stetig diffbaren Funktionen von Ω nach $\mathbb{R}^2$ auf einem Kompakten Träger von Ω .
Das Bild u ist eine Funktion von Ω nach $\mathbb{R}$, nach Definition gilt: $u \in BV(\Omega)$ falls $u \in L^1(\Omega)$ und $TV(u) = sup\{\int_\Omega u\, div(\phi)\, dx | \phi \in C_c^1(\Omega, \mathbb{R}^2), |\phi| \leqslant 1\} < +\infty$
wobei TV(u) Totalvariation von u genannt wird und manchmal als $\int_\Omega |\nabla u|$ geschrieben wird.
Eine messbare Menge $E \subset \Omega$ hat einen endlichen Umfang in Ω ,falls $\chi_E \in BV(\Omega)$;
Der Umfang P(E) von E ist wie folgt definiert:
$P(E) = TV(\chi_E)$

Ausserdem, wenn E eine Lipschitzkonstante besitzt gilt:
$P(E) = H^1(\partial E)$
Wenn $u \in BV(\Omega)$ für fast alle $\lambda \in \mathbb{R}$, hat E_λ einen endlichen Umfang und aus der Coarea Formel folgt:

$$TV(u) = \int_{-\infty}^{+\infty} P(E_\lambda)\, d\lambda \qquad (3.2)$$

Formel (1.3)

Das Modell:

Um (1.2) einen Sinn zu geben, nehmen wir an dass u stetig ist und integrieren (1.2) über den Levelsets E_λ .
Damit haben wir:

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \int_{E_\lambda} div\left(\frac{\nabla u}{|\nabla u|}\right) dx. \qquad (3.3)$$

Mit Gauss-Green und weil $\frac{\nabla u}{|\nabla u|}$ die äussere Normale η zu dem Niveau ∂E_λ ist, bekommen wir:

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \int_{\partial E_\lambda} \frac{\nabla u}{|\nabla u|} \cdot \eta \, dH^1 = H^1(\partial E_\lambda). \qquad (3.4)$$

Nun nehmen wir ein $u \in BV(\Omega)$ und weil, für fast alle $\lambda \in \mathbb{R}$, E_λ eine Menge mit beschränktem Umfang ist, vermuten wir, dass die Minimierung der Totalvariation von u uns zu einem Evolutionsgesetz für u führt, welches für fast alle $\lambda \in \mathbb{R}$ gegeben ist durch Formel (1.2):

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = P(E_\lambda). \qquad (3.5)$$

Bevor wir noch näher auf die Analysis der Gleichungen eingehen, wollen wir erst einmal betrachten wie diese sich in Bezug auf Bilddaten verhalten.
Angenommen wir halten die Variation der Grauwerte Konstant auf jeder zusammenhängenden Komponente der Level-Sets.
Für Bilder scheint dies eigentlich natürlich zu sein, aber nun stellen wir uns folgendes Beispiel vor:
Gegeben sei ein Bild u mit zwei Kreisscheiben auf denen u(x) =1 gilt, auf einem Hintergrund mit u (x) = 0.

Beispiel:
Seien nun $D_1 \, und \, D_2$ die Scheiben mit den Radien $r_1 \, und \, r_2 \, wobei \, r_1 < r_2$.
Wenn wir uns auf unser Modell berufen, folgt:

$$\forall x \in D_1 : u(t,x) = u(x) + t \frac{H^1(\partial D_1)}{|D_1|} = \frac{2t}{r_1}.$$

$$\forall x \in D_2 : u(t,x) = u(x) + t \frac{H^1(\partial D_2)}{|D_2|} = \frac{2t}{r_2}$$

$$\forall x \in \Omega \setminus (D_1 \cup D_2) : u(t,x) = u(x) = 1$$

Wir bekommen eine schrittweise Minderung des Kontrastes, wegen dem höheren Verhältnis $H^1(\partial D_1)/|D_1|$

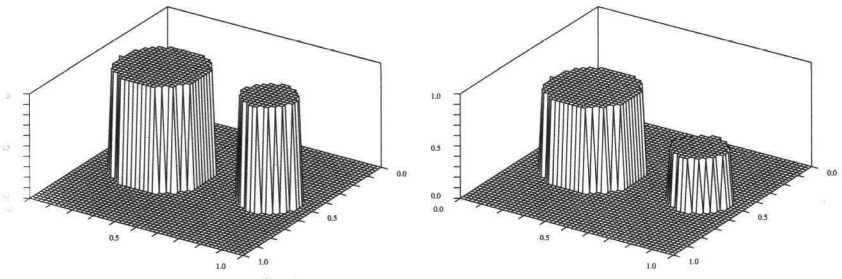

Abbildung 6

In Abbildung 6 links sehen wir das ursprüngliche Bild, und rechts das Bild nach einer Zeit t.

Der Wert des kleineren Kreises geht schneller gegen Null als der des grösseren Kreises.
Unter Umständen wird der kleinere Kreis vor dem grösseren Kreis ganz verschwinden.
Das entspricht dem zweiten Weg, die Totalvariation eines Bildes u zu verringern.

Die Gültigkeit des Modells für $u \in BV(\Omega) \cap C^{\infty}(\Omega)$.

Nun werden wir das Verhalten von (3.1) für reguläre Daten u genauer untersuchen.
Wir nehmen an, dass $u \in BV(\Omega) \cap C^{\infty}(\Omega)$ und benutzen fast dieselbe Idee wie im vorherigen Abschnitt, d.h. wir nehmen an, dass $\frac{\partial u}{\partial t}$ konstant ist auf jedem Niveau $\partial E_{\lambda} = \{x | u(x) = \lambda\}$ und zeigen, dass die Totalvarianz kleiner wird, wenn wir für solche u (3.1) anwenden.

Proposition:

Sei $u \in BV(\Omega) \cap C^{\infty}(\Omega)$ das sich nach (1.2) entwickelt.
Ausserdem nehmen wir an, dass $\frac{\partial u}{\partial t}$ konstant ist, auf jedem Niveau ∂E_{λ} , dann gilt:

$TV(u(t,x)) \leq TV(u(x))$ wobei TV(u(x)) die Totalvariation von u(x) ist.

Um diese Proposition beweisen zu können brauchen wir das folgende Lemma, in dem wir die Variation der Perimeter auf den Level Sets betrachten, für den Fall, dass u eine stetige Funktion ist.

Lemma:

Wenn u eine stetige Funktion ist, dann gilt für fast alle $\lambda \in \mathbb{R}$:

$$\frac{d}{d\lambda} H^1(\partial E_{\lambda}) = \int_{\partial E_{\lambda}} \frac{curv(u)}{|\nabla u|} dH^1.$$

Beweis des Lemmas:
Da u glatt ist, kann die Formel (1.2) geschrieben werden als:

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = H^1(\partial E_\lambda).$$

Ausserdem sagt uns Sards Theorem dass für fast alle $\lambda \in \mathbb{R}$ gilt:

$$\{|\nabla u| = 0\} \cap u^{-1}(\lambda) = \emptyset$$

Wenn man λ so wählt, dass Formel (3.2) erfüllt ist und h>0 klein genug, haben wir:

$$u(x + h\frac{\nabla u}{|\nabla u|^2}) = u(x) + h + o(h).$$

Wenn also x(s) eine Parametrisierung von ∂E_λ ist können wir formal annehmen, dass $\tilde{x}(s) = x(s) + h\frac{\nabla u}{|\nabla u|^2}$ die Parametrisierung von $\partial E_{\lambda+h}$ ist.

Nun schreiben wir der Einfachheit halber $x = (x_{1,} x_2).$ und nutzen die Tatsache, dass $\partial E_\lambda = \{(x_{1,} x_2) | u(x_{1,} x_2) = \lambda\}$

Dann haben wir $u_{x_1} x_1'(s) + u_{x_2} x_2'(s) = 0.$ Wenn wir jetzt ∂E_λ mit seiner euklydischen Bogenlänge parametrisieren, um $x_1'(s)^2 + x_2'(s)^2 = 1$ zu bekommen erhalten wir, wenn wir die Orientierung von ∂E_λ beachten,

$$x_1'(s) = -\frac{u_{x_2}}{|\nabla u|}$$

$$x_2'(s) = \frac{u_{x_1}}{|\nabla u|}.$$

Wir bezeichnen:

$$V = \frac{u_{x_1}}{|\nabla u|^2} \quad \text{und} \quad W = \frac{u_{x_2}}{|\nabla u|^2},$$

Daraus folgt dann:

$$\begin{aligned}
&(\tilde{x}_1'(s))^2 + (\tilde{x}_2'(s))^2 \\
&= 1 + \frac{2h}{|\nabla u|^2}(V_{x_1} u_{x_2}^2 - (V_{x_2} + W_{x_1}) u_{x_1} u_{x_2} + W_{x_2} u_{x_1}^2) + o(h) \\
&= 1 + \frac{2h}{|\nabla u|^4}(u_{x_1 x_1} u_{x_2}^2 - 2 u_{x_1 x_2} u_{x_1} u_{x_2} + u_{x_2 x_2} u_{x_1}^2) + o(h) \\
&= 1 + 2h\frac{curv(u)}{|\nabla u|} + o(h).
\end{aligned}$$

Somit erhalten wir:

$$H^1(\partial E_{\lambda+h}) = \int_0^{H^1(\partial E_\lambda)} \sqrt{(\tilde{x}_1'(s))^2 + (\tilde{x}_2'(s))^2}\, ds$$
$$= H^1(\partial E_\lambda) + h \int_{\partial E_\lambda} \frac{curv(u)}{|\nabla u|} dH^1 + o(h),$$

und schliesslich:

$$\frac{d}{d\lambda} H^1(\partial E_\lambda) = \int_{\partial E_\lambda} \frac{curv(u)}{|\nabla u|} d\, H^1.$$

Beweis der Proposition:

Wähle λ wie im Beweis des Lemmas und h>0 klein genug.
Und $E_\lambda = \{x | u(x) < \lambda\}$, haben wir für fast alle λ.

$$\begin{aligned} H^1(\partial E_{\lambda+h}) - H^1(\partial E_{\lambda-h}) &= \int_{E_{\lambda+h} \setminus E_{\lambda-h}} \frac{\partial u}{\partial t} dx \\ &= \int_{\lambda-h}^{\lambda+h} \left(\int_{\partial E_\nu} \frac{\partial u}{\partial t} \frac{1}{|\nabla u|} d\, H^1 \right) d\, \nu \\ &= 2h \int_{\partial E_\lambda} \frac{\partial u}{\partial t} \frac{1}{|\nabla u|} dH^1 + o(h). \end{aligned}$$

Weil wir angenommen haben dass $\frac{\partial u}{\partial t} = C_\lambda$ auf ∂E_λ , bekommen wir dann:

$$C_\lambda = \frac{\frac{d}{d\lambda} H^1(\partial E_\lambda)}{\int_{\partial E_\lambda} \frac{dH^1}{|\nabla u|}} = \frac{\int_{\partial E_\lambda} \frac{curv(u)}{|\nabla u|} dH^1}{\int_{\partial E_\lambda} \frac{1}{|\nabla u|} dH^1}.$$

Wir definieren $E_{t,\lambda} = \{x | u(t,x) < \lambda\}$ und $\partial E_{t,\lambda} = \{x | u(t,x) = \lambda\}$.

Dann betrachten wir x in ∂E_λ , t>0 klein genug und y so, dass

$$\vec{xy} = -t \frac{\partial u}{\partial t}(x) \frac{\nabla u}{|\nabla u|^2} = -tC_\lambda \frac{\nabla u}{|\nabla u|^2}.$$

Wir haben nun $u(t,y) = u(y) + t\frac{\partial u}{\partial t}(y) + o(t) = u(x) - t\frac{\partial u}{\partial t}(x) + t\frac{\partial u}{\partial t}(y) + o(t) = u(x) + o(t)$.

Wenn jetzt x(s) eine Parametrisierung von ∂E_λ ist, können wir annehmen, dass

$$y(s) = x(s) - tC_\lambda \frac{\nabla u}{|\nabla u|^2}$$

eine Parametrisierung von $\partial E_{t,\lambda}$ ist. Dadurch bekommen wir, wie im Beweis zum Lemma:

$$H^1(\partial E_{t,\lambda}) = H^1(\partial E_\lambda) - tC_\lambda \int_{\partial E_\lambda} \frac{curv(u)}{|\nabla u|} dH^1 + o(t),$$

und indem wir den Wert fü C_λ einsetzen:

$$H^1(\partial E_{t,\lambda}) = H^1(\partial E_\lambda) - t \frac{(\int_{\partial E_\lambda} \frac{curv(u)}{|\nabla u|} dH^1)^2}{\int_{\partial E_\lambda} \frac{1}{|\nabla u|} dH^1} + o(t).$$

Wenn wir uns auf die Coareaformel (3.2) berufen erhalten wir:

$$TV(u(t,x)) \leq TV(u(x)).$$

Damit ist der Beweis über die TV Minimierungseigenschaft von Formel (1.2) für ein regelmässiges u erbracht.

4.) Minimierung der Totalvariation mit FLST

Wir wollen die Totalvariation eines Bildes u reduzieren, indem wir die Shapes mit zu grosser Krümmung eliminieren. Anstatt dass wir die zusammenhängenden Komponenten der Levelsets betrachten, benutzen wir die Shapes die durch den FLST gebildet wurden.
Um schwarze Formen auf weissem Grund und weisse Formen auf schwarzem Grund auf die gleiche Weise behandeln zu können, müssen wir einen Operator $T_t(u)$ konstruieren, indem wir alternativ obere oder untere LevelSets betrachten.
Sei u nun ein Bild und $E_1, E_2, \dots, E_N = E_R$ die Shapes des FLST in umgekehrter Levelordnung, wobei E_R die Wurzel sei.
Wir bezeichnen den Grauwert der zu dem Shape E_i gehört mit λ_i .und mit $E_i(\leq \lambda_i)$ bzw. $E_i(\geq \lambda_i)$ die Tatsache, dass E_i aus einem unteren bzw. oberen Levelset besteht.
Es sei:

$$P(E_i) = H^1(\partial E_i \setminus \partial \Omega).$$

Dann kann die Coareaformel (3.2) wie folgt geschrieben werden:

$$TV(u) = \sum_{i=1}^{N-1} P(E_i)|\lambda_i - \lambda_i^p|, \qquad (4.1)$$

wobei λ_i^p der Gray Value ist, der zu E_i^p, dem Eltern-Shape von E_i , gehört.

Um (3.1) zu implementieren, oder genauer gesagt um für jeden Shape E_i

$\int_{E_i} \frac{\partial u}{\partial t} dx = P(E_i)$, zu implementieren,

bilden wir eine Folge von Bildern (u^m) so dass:

$$u^0 = u$$
$$u^m = T_t(u^{m-1}) = (T_t)^m(u), m \geq 1.$$

Definition des Operators T_t :
Für das Originalbild konstruieren wir eine endliche Folge $(u_0 = u, u_1, u_2, \dots, u_{N-1})$ wobei N die Anzahl der Shapes im FLST-Baum ist. Dann setzen wir:

$$T_t(u) = u_{N-1}.$$

Wobei die u_i wie folgt definiert werden:

Wenn u_{i-1} bekannt ist, erhalten wir u_i durch:

- wenn $x \in E_i$ und $E_i(\leq \lambda_i): u_i(x) = inf\{u_{i-1}(x) + t\frac{P(E_i)}{|E_i|}, \lambda_i^p\}$,

- wenn $x \in E_i$ und $E_i (\geq \lambda_i) : u_i(x) = sup\{u_{i-1}(x) - t\frac{p(E_i)}{|e-i|}, \lambda_i^p\}$,

- wenn $x \notin E_i$: $u_i(x) = u_{i-1}(x)$.

5.) Eigenschaften des Algorithmus

Minimierungs-Eigenschaften:

Nun wollen wir beweisen, dass die Totalvariation mit jedem Schritt kleiner wird, wenn wir den oben genannten Algorithmus verwenden.
$\lambda_i \geq \lambda_i^p$, dabei sei λ_i^p der Grauwerte der zu E_i^p , dem Elternshape von E_i , gehört. Seien $E_{1,} E_{2,...} E_j$ die Endknoten des Baumes und nehmen wir an, dass wir u_j bereits gebildet haben. Die Shapes, die unverändert geblieben sind bezeichnen wir mit $\lambda_{i,j}$ und $\lambda_{i,j}^p$ für die neuen Grauwerte zum Schritt j. Wir benutzen (4.1)um zu zeigen:

$$TV(u_j) = \sum_{i=1}^{N-1} P(E_j)|\lambda_{i,j} - \lambda_{i,j}^p| = \sum_{i=1}^{j} P(E_i)|\lambda_{i,j} - \lambda_i^p| + \sum_{i=j+1}^{N-1} P(E_j)|\lambda_i - \lambda_i^p|.$$

Wenn wir nur die Werte von u auf den Endknoten ändern bekommen wir in der Tat für $1 \leq i < N : \lambda_{i,j}^p = \lambda_i^p$; und für $j < i < N : \lambda_{i,j} = \lambda_i$.
Die neuen Werte $\lambda_{i,j}$ für $1 \leq i < j$ die durch die Konstruktion von T_t entstanden sind, sind identisch mit:

$$inf\{\lambda_i + t\frac{P(E_i)}{|E_i|}, \lambda_i^p\} \quad \text{oder mit} \quad sup\{\lambda_i - t\frac{P(E_i)}{|E-i|}, \lambda_i^p\},$$

je nachdem ob der Shape E_i ein oberer oder ein unterer Shape ist. Daraus schliessen wir:

$$TV(u_j) \leq TV(u) = TV(u_0).$$

Für den allgemeinen Fall nehmen wir ein Shape E_i mit $1 \leq i < N$. Der Grauwert der zu ihm gehört wird zuerst durch die Konstruktion von u_i verändert. Wir haben $|\lambda_{i,i} - \lambda_{i,i}^p| < |\lambda_{i,i-1} - \lambda_{i,i-1}^p|$ mit $\lambda_{i,i}^p = \lambda_{i,i-1}^p$ und $\lambda_{i,i}$ welches identisch zu

$$inf\{\lambda i, i-1 + t\frac{P(E_i)}{|E_i|}, \lambda_{i,i-1}^p\} \quad \text{oder} \quad sup\{\lambda_{i,i-1} - t\frac{P(E_i)}{|E_i|}, \lambda_{i,i-1}^p\}. \quad \text{ist.}$$

Untersuchen wir nun, wie die Grauwerte die zu dem Shape E_i gehören, von der weiteren Konstruktion von $T_t(u)$ modifiziert werden.

Während der Auswertung von $u_k, i < k < N$ begegnen wir zwei Fällen:

- $E_i \cap E_k = \emptyset$, dann ist $\lambda_{i,k-1} = \lambda_{i,k}$ *und* $\lambda_{i,k-1}^p = \lambda_{i,k}^p$.
- $E_i \subset E_k$, in diesem Fall haben wir:

$\lambda = inf\{\lambda_{i,k-1} + t\frac{P(E_k)}{|E_k|}, \lambda_{i,k-1}^p\}$ oder $sup\{\lambda_{i,k-1} - t\frac{P(E_k)}{|E_k|}, \lambda_{i,k-1}^p\}$, abhängig vom Typ des Shapes (oberer oder unterer). Nun, da der Elternshape von E_i auch in E_k enthalten sein wird,

wird der Wert $\lambda^p_{i,k-1}$ auf dieselbe Art wie $\lambda_{i,k-1}$ verändert und so haben wir wieder: $|\lambda_{i,k}-\lambda^p_{i,k}|=|\lambda_{i,k-1}-\lambda^p_{i,k-1}|$.

In jedem Schritt der Konstruktion der Folge $(u_k), 1\le k\le N-1$ haben wir dieselben Haupteigenschaften:

(i) Die Shapes die unverändert geblieben sind, können verschwinden, wenn $|\lambda_{i,k}-\lambda^p_{i,k}|=0$.
(ii) $|\lambda_{i,k-1}-\lambda^p_{i,k-1}|\le|\lambda_{i,k}-\lambda^p_{i,k}|$.
Daraus schliessen wir:
$TV(T_t(u))=TV(u_{N-1})\le TV(u)$.

Entrauschung und andere Eigenschaften:

Definieren wir die Krümmung einer Teilmenge C von Ω mit:
$curv(C)=\frac{P(C)}{|C|}$.
Nach Konstruktion verschwindet C, wenn die zugehörige Krümmung zu gross ist.
Die Ecken/Kanten von gut definierten, grossen Objekten bleiben erhalten. Das ist der Hauptunterschied zwischen unserem Modell und der 'Mean Curvature Motion' $\frac{\partial u}{\partial t}=|\nabla u|curv(u)$, oder der Erosion $\frac{\partial u}{\partial t}=|\nabla u|$, welche beide zu einer Minimierung der Totalvarianz von u führen. Ausserdem sind die Funktionen glatt. In unserem Modell bleibt eine grosse Fläche dieselbe, nur der Grauwert wird verändert.
Wenn wir annehmen, dass Rauschen nur einzelne Pixel, (oder kleine unregelmässige Regionen) betrifft, wird durch all diese Eigenschaften klargestellt, dass das Rauschen eliminiert wird. Ausserdem wird der TBLL nur verändert, indem Shapes weglassen werden, d.h. die Kanten der Shapes werden nicht verändert, daraus folgt, dass unsere Methode kantenerhaltend ist.

Es ist zu beachten, dass:
- Es kein Abbruchkriterium gibt, wie es normalerweise bei PDE-Modellen der Fall ist.
- Die Wahl der Zeitschritte t kontrolliert die Geschwindigkeit, mit der die Grauwerte, die zu den Shapes gehören, sich ändern.

6.) Beispiele

Vergleiche mit anderen Modellen und Algorithmen:

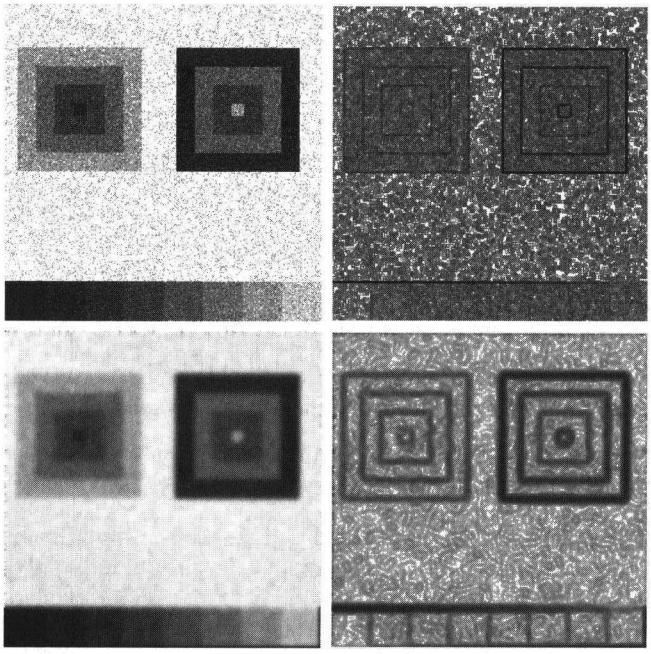

Abbildung 7

In Abbildung 7 kann man oben links das ursprüngliche verrauschte Bild u sehen, rechts sieht man den Gradienten des jeweiligen Bildes. In der unteren Zeile sieht, wie die Wärmeleitungsgleichung sich auf das Bild u auswirkt, man beachte die Dicke der jeweiligen Begrenzungslinien rechts.

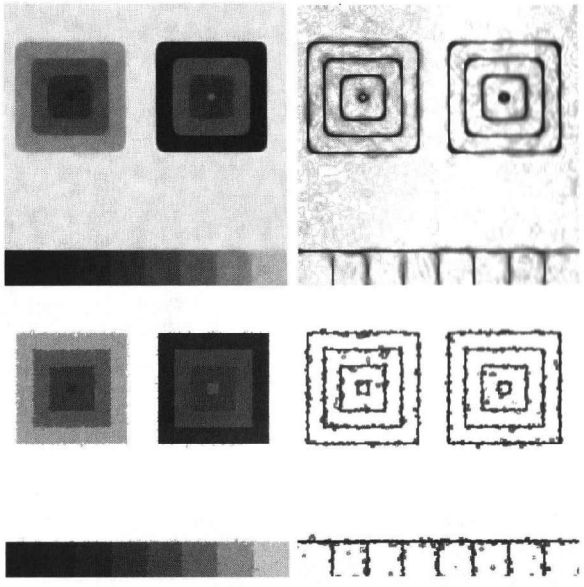

Abbildung 8

In Abbildung 8, oben wurde ein sogenannter MCM- Algorithmus auf das Bild angewandt. Man kann erkennen, dass dadurch die Ecken abgerundet worde sind
Unten ist das Ergebnis unseres Algorithmus abgebildet.

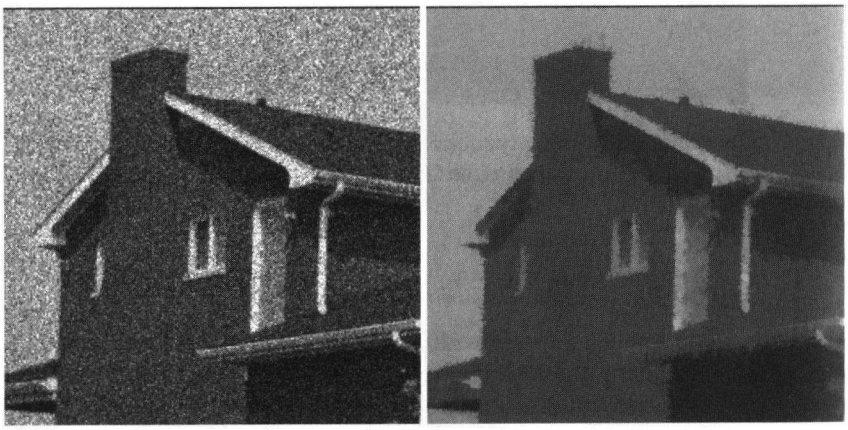

Abbildung 9

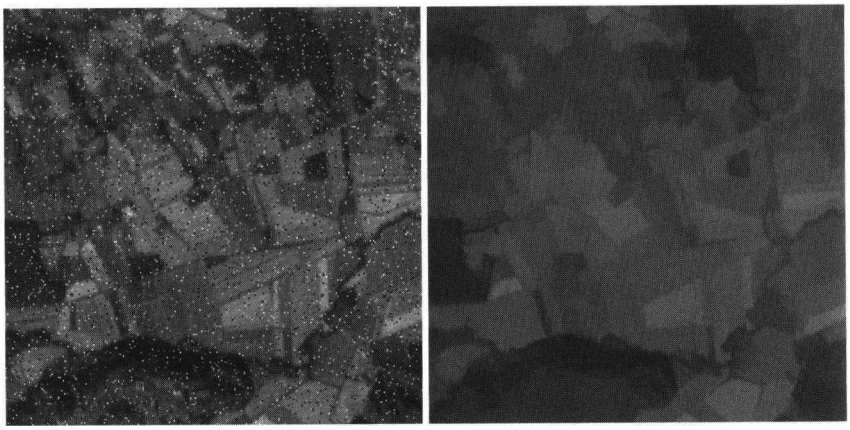

Abbildung 10

In Abbildung 9 und 10 kann man einen Vorher/Nachher Vergleich betrachten.

Farbbilder:

Die Entrauschung von Farbbildern ist in der Regel viel komplizierter, da ein geeigneter Farbraum gewählt werden muss. Der übliche RGB-Raum liefert oft schlechte Ergebnisse:
Wenn man auf jedem Farbkanal extra arbeitet, kann das Endbild unter Umständen sogenannte 'falsche' Farben enthalten, d.h Farben, die im Originalbild gar nicht enthalten waren.
Für unsere Zwecke ist es aber ausreichend auf dem RGB-System zu arbeiten. Ausserdem, wenn man den bilinearen FLST und eine geeignete Zeitschrittweite wählt kommt es nicht zu sichtbaren 'falschen' Farben.
Wir minimieren die totale Variation auf den 3 Kanälen R,G und B und konstruieren daraus ein entrauschtes Bild dessen Qualität sehr gut ist.
Dies führt uns zu der Annahme, dass wir zum Entrauschen nur denjenigen Kanal benutzen brauchen, der auch wirklich verrauscht ist, und den Algorithmus nicht auf alle Kanäle anwenden brauchen.

7.) Literatur

S.Osher, R.Paragios, eds. „Geometric level set methods in imaging, vision, and graphics“, Springer, 2005.