Brandenburgische Technische Universität Cottbus (BTU)

Lehrstuhl Praktische Informatik / Grafische Systeme

Studienarbeit

im Studiengang Informatik

Thema: Softwareentwicklung zur Identifikation und Vermessung von Jahrringen in digitalen Fotos von Stammscheiben

eingereicht von: Sebastian Puder <puder@tu-cottbus.de>

eingereicht am: 15. Oktober 2006

Betreuer: Herrn Prof. Dr. Winfried Kurth

1 Einleitung

Bäume können bei ungestörter Vegetation und unter sehr günstigen Verhältnissen ein außerordentliches Alter erreichen. Mit dem hohem Alter, das oft mehrere Jahrhunderte betragen kann, ist in der Regel eine ungewöhnliche Dicke des Stammes, aber nicht immer eine entsprechende Höhe verknüpft. Wächst der Baum unter im Jahresrhythmus schwankenden klimatischen Bedingungen, wird jedes Jahr ein Jahresring angelegt. Mit Hilfe dieser Ringe lässt sich das Alter eines solchen Baumes bestimmen und die Bedingungen in den einzelnen Jahren ablesen. Die Dendrochronologie nutzt dies, um altes Holz zu datieren und das Klima einer Region bis zu mehreren tausend Jahren zurückzuverfolgen. [bau06]
Die Dendrochronologie (griech. dendron = Baum, chronos = Zeit) ist eine Datierungsmethode der Archäologie, der Kunstwissenschaft und der Dendroökologie, bei der die Jahresringe von Bäumen gezählt werden. Der Begriff Dendrochronologie geht auf den amerikanischen Astronomen Andrew Ellicott Douglass (1867-1962) zurück. [den06]
Ziel und Zweck dieser Arbeit ist das Untersuchen, das Anwenden und die Auswertung bekannter Verfahren zur automatisierten Bildanalyse. Es werden 3 Verfahren vorgestellt, deren Mechanismen vorgestellt und auf Eignung zur Analyse von Stammscheiben untersucht. Zusätzlich ist eine Anwendung zu entwickeln, welche eine halbautomatisierte Analyse von Stammscheiben mittels verschiedener Verfahren vornimmt, um Informationen über das Alter des Baumes und über dessen Wachstum innerhalb einzelner Jahreszeiträume zu gewinnen. Verfahren, die in Betracht kommen für die Arbeit, sind morphologische Operatoren, Splines und Parabolic Blending. Die Realisierung der Arbeit erfolgt in der Programmiersprache Java und wird als Plugin für das Bildbearbeitungsprogramm ImageJ dienen.

2 Motivation

Anwendungen von Bildanalyseverfahren haben sich auf nahezu alle Ingenieur- und Wissenschaftsbereiche ausgebreitet: Visuelle Inspektion, Qualitäts- und Sicherheitskontrolle, Dokumentenverarbeitung, Geowissenschaften, Mikroskopie, Biologie und medizinische Bildbearbeitung. Diese zahlreichen Anwendungsgebiete haben zu einer großen Vielfalt an Bildanalyseproblemen geführt. Jeder Ansatz eignet sich sehr gut für eine bestimmte Problemart.

- Bildfilterung
- Bildsegmentierung
- Bildmessungen
- Interpolation von Konturdaten

Bisher wurde die Analyse von Stammscheiben interaktiv und teilweise manuell vorgenommen. Es soll versucht werden, nun moderne Mechanismen und Methoden auf die Analyse von Stammscheiben anzuwenden, um den sehr mühsamen und zeitaufwändigen Prozess der bisherigen Analyse entscheidend zu vereinfachen. [Soi98]

3 Der Baumring

Ein Jahresring spiegelt die verschiedenen Entwicklungsphasen in einer Vegetationsperiode wieder. Nach der Ruhephase im Winter (oder Trockenzeit) werden in der Mobilisierungsphase Nährstoffe verbraucht, die vor der Ruhephase angelegt wurden. Es folgt die Wachstumsphase, in der das sogenannte Frühholz entsteht. Es bildet sich eine relativ helle lockere Zuwachszone, die dem Baum den schnellen Transport von Wasser und Mineralien von der Wurzel in die Krone ermöglicht, um den Blattaustrieb und die Blütenbildung zu gewährleisten. Die Zellen im Frühholz sind dünnwandig und großlumig, dadurch mechanisch nicht sehr fest. In der darauf folgenden Depositionsphase entstehen dickwandige und kleinlumige Holzzellen, die wesentlich härter sind und hauptsächlich festigende Aufgaben übernehmen. In dieses Spätholz werden unterschiedliche Stoffe eingelagert, wodurch es eine dunklere Farbe bekommt. [jah06]

Abbildung 1: Stammscheibe

4 Morphologische Bildbearbeitung

4.1 Bildfilterung

Lokale Operatoren bzw. Punkt- und Nachbarschaftsoperatoren sind der Kernpunkt dieser Art der morphologischen Bildbearbeitung. Punktoperatoren verändern den Wert des Bildpunktes unabhängig von den Werten seiner Nachbarpunkte. Nachbarschaftsoperatoren oder auch lokale Operatoren genannt verändern den Wert des Punktes in Abhängigkeit von seinen Nachbarschaftspunkten. Gängige Beispiele in der Praxis sind: Unterdrückung von Rauschen, Kantenerkennung, Kompensation falscher Linseneinstellung und Unterdrückung von Bewegungsunschärfe. Für unser Projekt kommt die Kantenerkennung in Betracht, da sie eine der ersten Ideen der automatisierten Erkennung der Jahrringe war. Die Probleme, die mit diesem Verfahren auftraten, werden später erläutert. [Soi98]

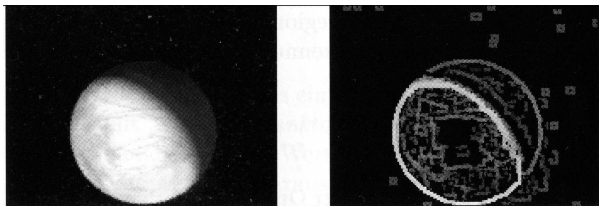

Abbildung 2: Kantenhervorhebung

4.2 Bildsegmentierung

Die Bildsegmentierung ist die Unterteilung eines Bildes in verschiedene Regionen, wobei jede Region bestimmte Eigenschaften besitzt. In einem segmentierten Bild sind die elementaren Bildelemente keine Pixel mehr, sondern eine verbundene Menge von Pixeln, die derselben Region angehören. Ist das Bild erst einmal segmentiert worden, kann jede Region ausgewertet werden. Bildsegmentierung spielt eine Rolle in der quantitativen Auswertung von Bilddaten. [Soi98]

4.3 Bildmessungen

Die Bildauswertung hat die Charakterisierung der Bildobjekte durch einige numerische Werte zum Ziel. Die Messung ist für ein gegebenes Kriterium unterschiedlich, wenn die Werte der Objekte, die dieses Kriterium erfüllen, sehr verschieden von denen aller anderen Objekte sind. Die Morphologie liefert eine breite Palette von Werkzeugen zur Bildmessung, wie zum Beispiel die Richtungs-, Textur- und Formanalyse. [Soi98]

5 Kantenerkennung

Die Kantenerkennung ist Teil einer Segmentierung in der Bildbearbeitung bei der versucht wird, flächige Bereiche in einem digitalen Bild von einander zu trennen. Es soll durch Kantenoperatoren erreicht werden, dass Übergänge zwischen Bereichen erkannt werden. Bei der Kantenerkennung werden Filtermasken (Matrizen in unterschiedlichen Größen) benutzt. Diese Maske wird dann auf ein Pixel und dessen angrenzende Pixel (nearest neighbor) angewandt und dann ein gewichteter Mittelwert berechnet (siehe Abb. 2). Je grösser eine Filtermaske ist, desto mehr Punkte werden in die Berechnung des Mittelwertes mit einbezogen. Bekannte Verfahren von Kantenfiltern sind:

- Sobel-Operator
- Laplace-Operator
- Prewitt-Operator
- Roberts-Operator
- Kirsch-Operator

In der Anwendung wird der Sobel-Operator benutzt, welcher bereits in ImageJ integriert ist und verwendet werden kann.

Filtermasken des Sobel-Operators: $S_x = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$ $S_y = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$

6 Splines

Spline ist die Bezeichnung für bestimmte Funktionen aus der Analysis mit Anwendungen in der numerischen Mathematik. Ein Spline n-ten Grades ist eine Funktion, die stückweise aus Polynomen mit maximalem Grad n zusammengesetzt ist. S heißt Splinefunktion der Ordnung k (oder vom Grad k-1), wenn gilt:

1. S ist in jedem Teilintervall $[t_r, t_{r+1}]$ ein Polynom vom Grad $k-1$
2. S und die Ableitungen S^m, $m = 1, \ldots, k-2$ sind stetig auf $[t_0, t_n]$
3. S heißt Sub-Splinefunktion, wenn (1) gilt, S stetig ist, (2) aber nicht gilt

Splines werden vor allem zur Interpolation benutzt, wo dann an den Verklebungsstellen Bedingungen gestellt werden, nämlich, dass der Spline (k-2) mal stetig differenzierbar ist. Durch die stückweise Definition wird der unangenehme Effekt der Polynominterpolation

beseitigt, dass für n+1 Stützpunkte ein Polynom n-ten Grades zur Interpolation benötigt wird. Dies führt unter anderem zu starken Abweichungen von der zu interpolierenden Funktion in den Randbereichen und zwischen den Stützpunkten (Oszillationen). Die Spline-Interpolation hat diese Probleme nicht. Sind die einzelnen Polynome alle linear, so nennt man den Spline linear (es handelt sich dann um einen Polygonzug), analog gibt es quadratische, kubische usw. Splines. Der Begriff stammt aus dem Schiffbau: eine lange dünne Latte (Straklatte, englisch spline), die an einzelnen Punkten durch Nägel fixiert wird, biegt sich genau wie ein kubischer Spline mit natürlicher Randbedingung. In unserem speziellen Anwendungsfall sollen die Splines Baumringe, anhand von gegebenen Stützstellen, so gut es geht interpolieren, um einen makellosen Verlauf der Baumringe darzustellen. [BK98] [spl06]

6.1 Kubische Splines

Ein kubischer Spline ist eine glatte Kurve, die durch gegebene Punkte im Koordinatensystem geht. Jedes Teilstück ist dabei durch eine kubische Parabel $a_i t^3 + b_i t^2 + c_i t + d_i$ mit geeigneten Koeffizienten a_i, b_i, c_i und d_i definiert. Zu jedem Abschnitt gibt es also 4 Unbekannte. Hat man n Abschnitte und $n+1$ Stützstellen, so hat man $4n$ Unbekannte, die man durch Lösen eines Gleichungssystems bestimmen muss.
Glatte Kurve bedeutet dabei im mathematischen Sinne, dass die Kurve zweimal stetig differenzierbar sein soll. Dadurch lassen sich Knicke an den Anschlussstellen vermeiden, welche bei linearen Splines auftreten würden. Alle gegebenen Punkte stellen als Stützstellen der Kurve auch Nahtstellen zwischen den Teilkurven dar, in denen jeweils beide Funktionswerte, beide erste und auch zweite Ableitungen der zusammentreffenden Teilkurven übereinstimmen müssen. Diese Naht- oder Stützstellen werden auch Knoten genannt.

Gesucht: Approximation einer Kurve $f(T)$, die durch n vorgegebene Punkte laufen soll mit Hilfe von $n-1$ Kurvenabschnitten. $T \in [t_0, t_n]$, $f(T) = (f_{x,i}(t), f_{y,i}(t))$

$$f_{x,i}(t) = \begin{cases} f_{x,1}(t), T \in [t_0, t_1], t = \frac{T-t_0}{t_1-t_0} \\ f_{x,2}(t), T \in [t_1, t_2], t = \frac{T-t_1}{t_2-t_1} \\ f_{x,3}(t), T \in [t_2, t_3], t = \frac{T-t_2}{t_3-t_2} \\ \vdots \\ \vdots \\ f_{x,n-1}(t), T \in [t_{n-1}, t_n], t = \frac{T-t_{n-1}}{t_n-t_{n-1}} \end{cases} \qquad f_{y,i}(t) = \begin{cases} f_{y,1}(t), T \in [t_0, t_1], t = \frac{T-t_0}{t_1-t_0} \\ f_{y,2}(t), T \in [t_1, t_2], t = \frac{T-t_1}{t_2-t_1} \\ f_{y,3}(t), T \in [t_2, t_3], t = \frac{T-t_2}{t_3-t_2} \\ \vdots \\ \vdots \\ f_{y,n-1}(t), T \in [t_{n-1}, t_n], t = \frac{T-t_{n-1}}{t_n-t_{n-1}} \end{cases}$$

Bemerkung: f wird in parametrisierter Form ermittelt (d.h. $x = f_x(t), y = f_y(t)$), da die explizite Form $y = f(x)$ pro x-Wert nur einen y-Wert erlaubt.

$$x = f_{x,i}(t) = a_{x,i} t^3 + b_{x,i} t^2 + c_{x,i} t + d_{x,i}, t \in [0, 1]$$

$$x = f_{x,i}(t) = (((a_{x,i}t) + b_{x,i})t + c_{x,i})t + d_{x,i}$$
$$y = f_{y,i}(t) = a_{y,i}t^3 + b_{y,i}t^2 + c_{y,i}t + d_{y,i}, t \in [0,1]$$
$$y = f_{y,i}(t) = (((a_{y,i}t) + b_{y,i})t + c_{y,i})t + d_{y,i}$$

Es gibt mehrere Möglichkeiten den Parameter t zu wählen. Eine Möglichkeit ist die, in der t den Abstand zwischen den gegebenen Punkten P_i und P_{i+1} repräsentiert.

$$t_{i,max} = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

Die andere Möglichkeit ist die, wobei der Abstand zwischen den Punkten P_i und P_{i+1} künstlich auf 1 normiert wird.

$$t \in [0,1]$$

Ich verwende die letztere Methode, weil spätere Berechnungen dadurch vereinfacht werden und ich keine zusätzlichen Werte zwischenspeichern muss.
Die Formel $f_i(t) = a_i t^3 + b_i t^2 + c_i t + d_i$ enthält die noch frei wählbaren Koeffizienten a_i, b_i, c_i und d_i. Für die Bestimmung der kubischen Spline-Funktion $f_i(t)$ sind somit $4n$ Koeffizienten zu berechnen. Dafür stellt man aus $4n$ Bedingungen ein lineares Gleichungssystem bezüglich dieser Unbekannten auf und löst es zum Beispiel mit dem Gauss-Verfahren. Durch die besondere Struktur des Ansatzes $f_i(t)$ wird diese Aufgabe erleichtert. Die $4n$ Bedingungen resultieren aus speziellen geometrischen Eigenschaften, die der Spline $f_i(x)$ besitzen soll. Hauptsächlich sind dies Eigenschaften in den inneren Stützsstellen, wo der Spline von $f_i(t)$ in $f_{i+1}(t)$ übergeht. Man fordert die Erfüllung der folgenden $4n$ Bedingungen bzw. Gleichungen.

Randbedingungen:

Interpolationseigenschaften ($2n$ Bedingungen)

$f_i(1) = f_{i+1}(0)$
$f_i(0) = d_i$
in jedem linken Randpunkt: $d_{x,i} = x_i$ ($d_{y,i} = y_i$)
$f_i(1) = d_{i+1}$
in jedem rechten Randpunkt: $a_i + b_i + c_i + d_i = d_{i+1}$

Stetigkeit der Tangente ($n-1$ Bedingungen)

in jeder inneren Stützstelle: $f_i^{'}(1) = f_{i+1}^{'}(0)$
$3a_i + 2b_i + c_i = c_{i+1}$

Stetigkeit der Krümmung ($n-1$ Bedingungen)

in jeder inneren Stützstelle: $f_i^{''}(1) = f_{i+1}^{''}(0)$
$6a_i + 2b_i = 2b_{i+1}$

natürliche Randbedingungen (2 Bedingungen)

an den Randpunkten des Intervalls: $f_0^{''}(0) = 0$, $f_{n-1}^{''}(1) = 0$

(dies ist die Randbedingung für offene, kubische Splines, für geschlossene, kubische Splines sieht die Randbedingung wie folgt aus: $f_0^{'}(0) = f_{n-1}^{'}(1)$ und $f_0^{''}(0) = f_{n-1}^{''}(1)$)

Bestimmen der Ableitungen:

1. $f_i(t) = a_i t^3 + b_i t^2 + c_i t + d_i$
2. $f_i^{'}(t) = 3a_i t^2 + 2b_i t + c_i$
3. $f_i^{''}(t) = 6a_i t + 2b_i$

Bestimmung der Koeffizienten:

- $f_i(0) = d_i = x_i$
- $f_i(1) = a_i + b_i + c_i + d_i = d_{i+1}$
- $f_i^{'}(0) = c_i = D_i$ (An dieser Stelle füge ich die neue Variable D_i hinzu, um weiterführende Schritte übersichtlicher zu gestalten. (D steht für Derivation, also Ableitung.))
- $f_i^{'}(1) = 3a_i + 2b_i + c_i = D_{i+1}$

1. $a_i = 2(x_i - x_{i+1}) + D_i + D_{i+1}$
2. $b_i = 3(x_{i+1} + x_i) - 2D_i - D_{i+1}$
3. $c_i = D_i$
4. $d_i = x_i$

Wie zu erkennen ist, gibt es noch die Unbekannte D_i in diesen Gleichungen, um die Koeffizienten berechnen zu können. Um die D_i zu berechnen ist es zusätzlich erforderlich, dass die zweiten Ableitungen an den Innenpunkten zusammenpassen (Differenzierbarkeit),

- $f_{i-1}(1) = x_i$
- $f'_{i-1}(1) = f_i(0)$
- $f_i(0) = x_i$
- $f''_{i-1}(1) = f''_i(0)$

als auch, für die Endpunkte,

- $f_0(0) = x_0$
- $f_{n-1}(1) = x_n$

Bei offenen, kubischen Splines erhält man zu n gegebenen Punkten eine Gesamtmenge der Gleichungen von $4(n-2)+2$ für die $4(n-1)$ Unbekannten. Der Grund ist der, dass nur $n-1$ Subsplines berechnet werden. Um die zwei weiteren Bedingungen zu erreichen, ist es nun erforderlich, dass die zweiten Ableitungen an den Endpunkten NULL sind. Dagegen bei geschlossenen, kubischen Splines müssen zu n gegebenen Punkten n Subsplines berechnet werden. Das bedeutet, dass an Anfangs- und Endpunkt die gleichen Bedingungen gelten, wie zwischen den anderen Punkten, da das System geschlossen ist. Daraus folgt, dass man zu n gegebenen Punkten auch $4n$ Gleichungen erhält.

- $f''_0(0) = 0$ $(f''_0(0) = f''_{n-1}(1))$
- $f''_{n-1}(1) = 0$ $(f'_0(0) = f'_{n-1}(1))$

Es erfolgt ein Neuordnen dieser Gleichungen. Dabei werden die Ableitungen gebildet und ineinander eingesetzt. Als Resultat erhält man die nachfolgende Gleichung.

$$t_{i+2}D_i + 2(t_{i+2} + t_{i+1})D_{i+1} + t_{i+1}D_{i+2} = \frac{3}{t_{i+1}t_{i+2}} * (t^2_{i+1}(x_{i+2} - x_{i+1}) + t^2_{i+2}(x_{i+1} - x_i))$$

Aufgrund der Wahl unseres t Parameters und der Tatsache, dass wir einen normierten Abstand von 1 gewählt haben, können wir die t_i's in der obigen Formel sehr schnell ersetzen. Nach dem Einsetzen ergibt sich folgende Formel:

$$D_i + 4D_{i+1} + D_{i+2} = 3(x_{i+2} - x_i)$$

Daraus ableitend kann man ein symmetrisches, tridiagonales System herleiten. [1]

[1] Eine tridiagonale Matrix ist eine Matrix, deren Elemente 0 sind, ausser auf der Hauptdiagonalen und die Elemente darüber und darunter

Für offene, kubische Splines ergibt sich folgende Matrix. Aufgrund der Randbedingungen $f_0''(0) = 0$, $f_{n-1}''(1) = 0$ ergibt sich für D_0 und D_n eine besondere Form.

$$\begin{pmatrix} 2 & 1 & & & & & \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & 1 & 4 & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & 1 & 4 & 1 \\ & & & & & 1 & 2 \end{pmatrix} \begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_{n-1} \\ D_n \end{pmatrix} = \begin{pmatrix} 3 \cdot (x_1 - x_0) \\ 3 \cdot (x_2 - x_0) \\ 3 \cdot (x_3 - x_1) \\ \vdots \\ 3 \cdot (x_{n-1} - x_{n-3}) \\ 3 \cdot (x_n - x_{n-2}) \\ 3 \cdot (x_n - x_{n-1}) \end{pmatrix}$$

Für geschlossene kubische Splines ändern sich die Randbedingungen, so dass die Matrix für geschlossene, kubische Splines wie folgt aussieht.

$$\begin{pmatrix} 4 & 1 & & & & & 1 \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & 1 & 4 & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & 1 & 4 & 1 \\ 1 & & & & & 1 & 4 \end{pmatrix} \begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_{n-1} \\ D_n \end{pmatrix} = \begin{pmatrix} 3 \cdot (x_1 - x_n) \\ 3 \cdot (x_2 - x_0) \\ 3 \cdot (x_3 - x_1) \\ \vdots \\ 3 \cdot (x_{n-1} - x_{n-3}) \\ 3 \cdot (x_n - x_{n-2}) \\ 3 \cdot (x_0 - x_{n-1}) \end{pmatrix}$$

Die Matrix ist eine nxn-Matrix. Ein lineares Gleichungssystem A $\mathbf{x} = \mathbf{b}$ besitzt genau eine Lösung, wenn $\mathrm{Rg}A = \mathrm{Rg}(A, \mathbf{b}) = n$. Aufgrund der Struktur der Matrix ist zu erkennen, dass es zu keinen linearen Abhängigkeiten kommt. Anhand der Determinante kann man feststellen, ob das Gleichungssystem eine eindeutige Lösung besitzt (dies ist genau dann der Fall, wenn die Determinante ungleich Null ist). Nach Berechnung der Ableitungen D_i können die Koeffizienten a_i, b_i, c_i und d_i ermittelt werden.

Umsetzung im Programm

Im Programm bekommt die Funktion *Cubic[] calcClosedSpline(int n, double[] x)* die Anzahl der eingegebenen Stützpunkte und das jeweilige Array mit x- bzw. y-Koordinaten übergeben. Da die Matrix von der Struktur starr ist, nur in der Grösse variabel, kann ich die Matrix je nach Anzahl der gegebenen Punkte skalieren. Als erstes wird die rechte Seite der Matrix berechnet. Dazu werden jeweils die x- bzw. die y-Koordinaten benötigt, die der Funktion vorher übergeben wurden. Danach wird die Matrix mit den errechneten Werten gefüllt und mittels Gauss auf Zeilen-Stufenform gebracht. Jetzt können die gesuchten Werte D_i berechnet werden. Danach werden die Koeffizienten a_i, b_i, c_i, d_i berechnet und durch die folgende Struktur gespeichert. Diese Berechnung wird 2-mal durchgeführt, einmal um alle gesuchten Koeffizienten der x-Koordinaten und einmal um alle gesuchten Koeffizienten der y-Koordinaten zu ermitteln. [cur06] [con] [spla] [kub00]

[cub06] [splb] [kub]

```
public class Cubic {

    double a,b,c,d;

    public Cubic(double a, double b, double c, double d){
        this.a = a;
        this.b = b;
        this.c = c;
        this.d = d;
    }

    public double calc(float u) {
         /* a*u^3 + b*u^2 + c*u + d */
         return (((a*u) + b)*u + c)*u + d;
    }
}
```

Nachdem die Koeffizienten a_i, b_i, c_i, d_i ermittelt und gespeichert worden sind, wird nun der Spline gezeichnet. Je nach Wahl des Definitionsbereiches des Parameter t wird in einer einfachen Schleife in einer gewissen Anzahl von Schritten der Parameter t erhöht und die jeweilige x- bzw. y-Koordinate mittels der Funktion *double calc(float t)* berechnet.

```
    for (int j = 1; j <= STEPS; j++) {

        float u = j / (float) STEPS;

        int newPixelX = (int)Math.round(X[i].calc(u));
        int newPixelY = (int)Math.round(Y[i].calc(u));

        ip.putPixel(newPixelX,newPixelY,PixelColor);
    }
```

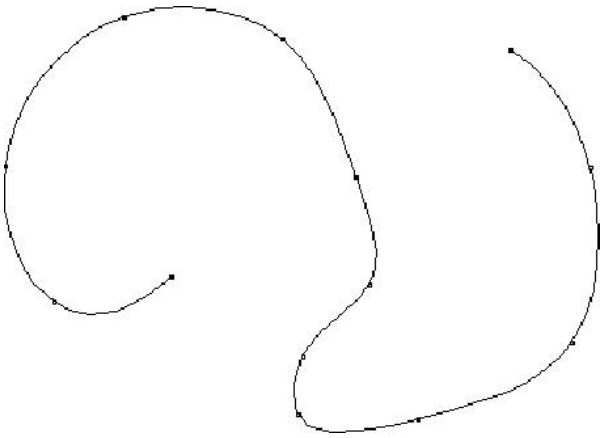

Abbildung 3: durch den Algorithmus erzeugter natürlicher Spline

6.2 B-Splines

Wie auch der Raum der Polynome ist der Raum der stückweisen Polynome ein Vektorraum und hat eine Basis. Im Kontext numerischer Verfahren, wo Splines häufig eingesetzt werden, ist die Wahl der Basis entscheidend für eventuelle Rundungsfehler und damit für die praktische Einsetzbarkeit. Eine bestimmte Basis hat sich hier als am besten geeignet herausgestellt: sie ist numerisch stabil und erlaubt die Berechnung von Werten der Spline-Funktion mittels einer Drei-Term-Rekursion. Diese so genannten B-Spline-Basisfunktionen haben einen kompakten Träger, sie sind nur auf einem kleinen Intervall von Null verschieden. Änderungen der Koeffizienten wirken sich also nur lokal aus. Splines, die in dieser Basis dargestellt werden, nennt man B-Splines.

Bei der Realisierung der Arbeit greife ich auf kubische Splines zurück, weil diese für meine Anwendung als geeigneter erscheinen als B-Splines. Der Hauptgrund ist, dass bei kubischen Splines die interpolierte Kurve durch jede der gegebenen Stützstellen verläuft, im Gegensatz zu den B-Splines. (siehe Abb. 4)

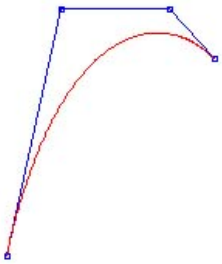

Abbildung 4: B-Spline mit n=4

7 Parabolic Blending

Parabolic Blending ist ein Verfahren, um eine knickfreie Kurve S durch eine Folge von Punkten zu legen. Man kann es je nach Problemstellung als explizite Darstellung y=y(x) oder als Parameterdarstellung $x = x(t)$,$y = y(t)$ verwenden. In jedem Intervall zwischen zwei Punkten Nr. i und i+1 werden zwei quadratische Polynomkurven gebildet. Eine davon geht durch die 3 Punkte mit den Indizes $i-1$, i, $i+1$, die andere durch die 3 Punkte mit den Indizes i, $i+1$, $i+2$. Wie der Name schon sagt, bedient sich Parabolic Blending dem Konzept der Nutzung von Blending Funktionen, um 2 parametrisch definierte Polynome 2. Grades in ein kubisches Polynom über einem bestimmten Intervall zu überführen.

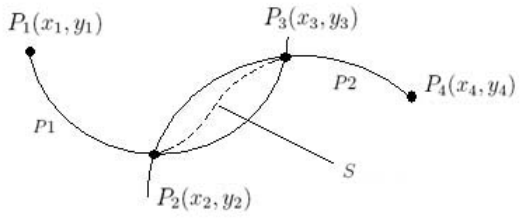

Abbildung 5: Parabolic Blending

Die explizite Form für Interpolationspolynome nach Lagrange sieht wie folgt aus.

$$P(x) = \sum_{i=1}^{n} \left(f(x_i) * \prod_{\substack{j=1 \\ i \neq j}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \right)$$

parametrische Form für Parabeln nach Lagrange für n=3

$$P_x(t) = \sum_{i=1}^{3} \left(x_i * \prod_{\substack{j=1 \\ i \neq j}}^{3} \left(\frac{t - t_j}{t_i - t_j} \right) \right), t \in [0, 1]$$

$$P_x(t) = \frac{(t-t_2)(t-t_3)}{(t_1-t_2)(t_1-t_3)} x_1 + \frac{(t-t_1)(t-t_3)}{(t_2-t_1)(t_2-t_3)} x_2 + \frac{(t-t_1)(t-t_2)}{(t_3-t_1)(t_3-t_2)} x_3$$

$$P_y(t) = \sum_{i=1}^{3} \left(y_i * \prod_{\substack{j=1 \\ i \neq j}}^{3} \left(\frac{t - t_j}{t_i - t_j} \right) \right), t \in [0, 1]$$

$$P_y(t) = \frac{(t-t_2)(t-t_3)}{(t_1-t_2)(t_1-t_3)} y_1 + \frac{(t-t_1)(t-t_3)}{(t_2-t_1)(t_2-t_3)} y_2 + \frac{(t-t_1)(t-t_2)}{(t_3-t_1)(t_3-t_2)} y_3$$

Wie in Abb. 5 dargestellt, werden 2 Parabeln für die Berechnung von S benötigt. Die Gleichungen für die Parabeln P1 und P2 werden nach der gerade erläuterten Vorschrift ermittelt. Die gesuchte Kurve S ist definiert als gewichtetes Mittel der Parabeln P1 und P2.

$$S_x(t) = (1-t) \cdot P1_x \left(\frac{t+1}{2} \right) + t \cdot P2_x \left(\frac{t}{2} \right), t \in [0, 1]$$

$$S_y(t) = (1-t) \cdot P1_y \left(\frac{t+1}{2} \right) + t \cdot P2_y \left(\frac{t}{2} \right), t \in [0, 1]$$

t ist für die Gewichtung der Kurve S zuständig. Das bedeutet, dass die Kurve sich Schritt für Schritt vom Kurvenverlauf der Parabel P1 dem Kurvenverlauf der Parabel P2 nähert, zwischen den Punkten P_2 und P_3. Es ist gleichbedeutend mit einer prozentualen Steigerung bzw. Annäherung des Kurvenverlaufes von Parabel P1 an die Parabel P2.
Eigenschaften:

- Die parabolischen Blendingfunktionen sind Subspline-Funktionen dritten Grades
- leicht zu programmieren
- lokale Interpolation, Oszillation gering [Kur04]

Umsetzung im Programm

Für die Umsetzung dieses Verfahrens zur Subsplineinterpolation werden die x- und y-Koordinaten in einem speziellen Format verwaltet. Die x- und y-Koordinaten werden jeweils in einem Vektor vom Typ $FollowPoints$ abgelegt, welcher zusätzlich zu jeder x- und y-Koordinate die zwei x- und y-Folgekoordinaten hinzugefügt bekommt.

```
public class FollowPoints {
    double P1;
    double P2;
    double P3;

    public Point(double P1, double P2, double P3){
        this.P1 = P1;
        this.P2 = P2;
        this.P3 = P3;
    }
}
```

Die in dieser Struktur gespeicherten Daten lassen sich nun sehr einfach zur Berechnung des Subsplines benutzen. Jedes Element des Vektors beinhaltet die 3 Koordinaten, die benötigt werden, um eine Parabelgleichung nach obiger Form aufzustellen. Jetzt werden die Elemente paarweise ausgelesen und aus den Koordinaten jedes Elementes 2 Parabelgleichungen aufgestellt. Mittels dieser 2 Parabelgleichungen wird die Gleichung der jeweils gesuchten Kurve S in diesem Abschnitt gebildet und die jeweiligen Koordinaten der gesuchten Kurve S berechnet. Die letzten beiden Paarungen sind das letzte und das erste Element. Die gesuchte Kurve S, die mittels dieser beiden Parabeln ermittelt wird, sorgt dafür, dass der Spline geschlossen wird.

8 Umsetzung

Die Anwendung hatte die Aufgabe, Kanten zu erkennen, diese zu verfolgen, um somit den Kreis bzw. die Jahrringe zu finden. Es sollte also eine automatische Analyse des Bildes vorgenommen werden, welche selbstständig die Kanten entdecken sollte. Zu diesem Zweck überprüfte ich die Eignung von morphologischen Operatoren. Die Qualität der Analyse hängt allerdings sehr von der Qualität der Stammscheibe ab, ob die Jahrringe, Früh- und Spätholz gut erkennbar sind. Zur Erkennung von Kanten wurden morphologische Operatoren benutzt, welche die automatische Erkennung der Kanten vornahmen. Nachdem die Kantenerkennung erfolgt war, wurde entlang eines bestimmten Bereiches nach Änderungen in der Helligkeitsverteilung gesucht, die einen Aufschluss darauf geben sollte, ob ein Jahrring vorliegt oder nicht. Probleme treten hier auf, wenn die Kanten der Jahrringe sehr schlecht erkennbar sind. Dadurch kann es bei der Analyse zu einer sehr hohen Fehlerquote kommen, welches die Folge hat, dass Ringe gar nicht erkannt werden. Als Folge konnte auch keine effektive Kantenverfolgung vorgenommen werden. Da die automatisierte Erkennung erhebliche Probleme aufwies, musste eine andere Möglichkeit gefunden werden. Als ein alternatives Verfahren empfohlen sich Splines, welche anhand

von gegebenen Stützstellen den Jahrring interpolieren sollten. Die Idee war hier, dass der Benutzer des Programms entlang eines für ihn sichtbaren Jahrringes Stützpunkte festlegen sollte, welche dann durch einen Algorithmus interpoliert werden sollten und hoffentlich genau den Verlauf des Jahrringes treffen. Als Interpolationsalgorithmuen werden hier geschlossene, kubische Splines oder Parabolic Blending Kurven benutzt, welche dafür sorgen, dass der Ring am Start- und Endpunkt geschlossen ist. Das sorgt dafür, dass ein Kurvenverlauf erzeugt wird, der keine Sprünge und Lücken aufweist.

Abbildung 6: Einen Jahrring interpolierende Splinekurve

Die Ergebnisse des Verfahrens sind in Abb. 6 exemplarisch dargestellt. Wie zu sehen ist, ist die Annäherung der Kurve an d en Jahrring gut bis sehr gut, weist allerdings unter bestimmten Gesichtspunkten dennoch einige Schwächen auf. Ein großes Problem besteht dann, wenn Stützpunkte sehr dicht beieinander liegen. Hierbei kommt es in der Regel zu Oszillationen, welche den makellosen Kurvenverlauf stören. (siehe Abb. 7) Auf der Suche nach der Ursache vermutete ich zuerst, dass sich ein Fehler in die Implementierung eingeschlichen hatte. Also untersuchte ich die Problematik etwas genauer und stellte fest, dass die Ursache dieser Oszillationen ein Problem der Interpolation war und kein Fehler in der Implementierung. Die Stützpunkte, die gegeben waren, werden nämlich von dem Algorithmus nicht als Extrema, Minimal- bzw. Maximalwerte interpretiert. Das bedeutet also, dass die Oszillation eine Schwachstelle des Algorithmus ist und dann auftritt, wenn Punkte sehr dicht beieinander liegen. Auf der Suche nach einer Lösung des Problems der Oszillationen probierte ich ein anderes Verfahren aus, Parabolic Blending.
Dieses Verfahren unterscheidet sich etwas von dem Verfahren der kubischen Splines. Man bestimmt für je 3 aufeinanderfolgende Punkte das Lagrange-Polynom 2. Grades (Parabel) und bildet zwischen je 2 Punkten eine gewichtete Mittel-Kurve S aus den beiden überlappenden Parabeln.

In der Tat tritt hier die Problematik mit den Oszillationen bei sehr dicht zusammen

liegenden Punkten nicht so stark auf wie bei den kubischen Splines. Aufgrund der unterschiedlichen Ergebnisse der Untersuchungen habe ich beide Verfahren in das Projekt implementiert, um je nach Fall den geeigneten Algorithmus wählen zu können.
Abschliessend bleibt zu sagen, dass nicht erwartet werden sollte, dass der Splinealgorithmus dafür sorgt, dass der erzeugte Spline exakt auf dem Jahrring liegt. Ferner soll und muss an den eingegebenen Stützstellen experimentiert (das bedeutet, dass die Stützstellen mal verschoben werden sollen) und gegebenfalls Stützpunkte hinzugenommen werden, um ein besonders genaues Ergebnis zu erreichen.

Abbildung 7: Oszillationen

9 Vergleich der Verfahren

Morphologische Operatoren:

Vorteil:

- gute Erkennung von Jahrringen bei sehr guten Stammscheiben

Nachteil:

- sehr schlechte Erkennung schon bei minimal fehlerhaften Stammscheiben
- großer Rechenaufwand bei sehr großen Bildern
- Kantenverfolgung schlägt fehl, wenn die Stammscheibe einige Verunreinigungen aufweist (z.B. Äste)
- Aufgrund der unterschiedlichen Wachstumsperioden des Baumes sind Früh- und Spätholz verschieden ausgeprägt, was zu Problemen bei der Kantenerkennung führen kann

Anmerkung: Morphologische Operatoren werden zwar verwendet, indem die Möglichkeit geboten wird, die Kantenerkennung auf das Ursprungsbild anzuwenden, allerdings verzichte ich auf eine automatisierte Analyse des Bildes. Der Grund ist, dass ich der Auffassung bin, dass bei einer automatisierten Analyse zu viele Faktoren für eine erfolgreiche Erkennung eine Rolle spielen. Kleinste Fehler oder Ungenauigkeiten in der Stammscheibe können zu einer fehlerhaften Kantenerkennung und -verfolgung führen.

Kubische Splines:

Vorteil:

- Parametrisierte Darstellungen erlauben Berechnungen unabhängig von Koordinatensystem
- Manuelle Eingabe der Stützstellen sorgt für eine genaue Erkennung der Jahresringe

Nachteil:

- Oszillationen bei dichter Stützstellenkonstellation
- Dadurch, dass die Stützstellen nicht als Hoch- oder Tiefpunkte interpretiert werden, kommt es teilweise zu größeren Oszillationen
- Hoher Rechenaufwand bei Bildern in sehr hoher Auflösung

Parabolic Blending:

Vorteil:

- Parametrisierte Darstellungen erlauben Berechnungen unabhängig von Koordinatensystem
- Manuelle Eingabe der Stützstellen sorgt für eine genaue Erkennung der Jahresringe
- Geringe Oszillationen bei sehr eng zusammen liegenden Punkten
- leicht zu implementieren

Nachteil:

- hoher Rechenaufwand bei Bildern in sehr hoher Auflösung
- die Bogenlänge ist geringer, im Gegensatz zu den Kurven, die mittels kubischer Splines erzeugt wurden

Abbildung 8: kubischer Spline

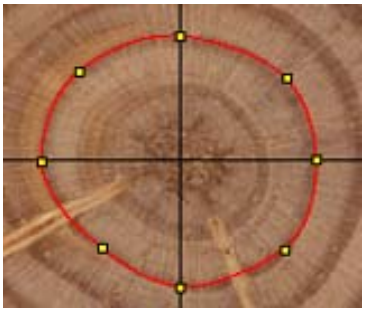

Abbildung 9: Parabolic Blending

10 Die Exportfunktion

Ein weiterer und wesentlicher Bestandteil der Anwendung soll die Exportfunktion sein. Diese soll die gesammelten und angefallenen Daten aufbereiten und eine weitere Bearbeitung durch das in Göttingen befindliche Analysesystem erlauben. Dazu werden die ermittelten Informationen aufbereitet und in das entsprechende Format gebracht, so dass diese vom Analysesystem in Göttingen verwendet werden können. Die Exportdatei beinhaltet sämtliche Daten, die während der Jahrringvermessung angefallen sind. Dazu gehören Grösse und Maßstab der Vorlage (der Maßstab muss von Hand eingegeben werden), sowie die Stützpunkte, die der Benutzer manuell angegeben hat. Die Struktur der Exportdatei ist relativ einfach aufgebaut. Sie ist in 3 Teile gegliedert.

Der erste Teil beschreibt die wesentlichen Eigenschaften der Bildvorlage, d.h. die entsprechenden Bildkoordinaten und den Maßstab (bzw. Umrechnungsfaktor).

```
*  **  GRAFIN - Grafik-Initialisierung                    (Zeile 1)
*  Ecke links unten    <xdat>   <ydat>
*  Ecke rechts oben    <xdat>   <ydat>
*  Umrechnungsfaktor HERCULES       <xfaktor>             (Zeile 4)
```

Im zweiten Teil werden die Stützstellen, die der Nutzer manuell eingegeben hat, jedem Jahring (Bsp. 1. Messung entspricht dem 1. Jahrring, 2. Messung entspricht 2. Jahrring, ...) zugeordnet. Nachfolgend zu jedem Jahrring (Messung) sind die Koordinaten der Stützstellen angegeben.

```
**  MASTAB  --  Koordinaten der Maßstabspunkte:          (Zeile 5)
*      1. Messung, 1. Punkt
*        x >......   y >......
*      1. Messung, 2. Punkt
*        x >......   y >......
*      :
*      :
*      3. Messung, 3. Punkt
*      x >......   y >......                              (Zeile 23)
```

Der dritte Teil beschreibt die Koordinaten des Mittelpunktes der Stammscheibe.

```
*  **  MITPKT - Mittelpunkt der Stammscheibe               (Zeile 24)
*        x >......   y >......                             (Zeile 25)
```

Es ist zur Zeit nicht möglich, anhand dieses Plugins eine komplette, exportfähige Datei zu erzeugen. Es gibt Parameter, die nur durch das in Göttingen befindliche System gegeben sind, welche nicht reproduzierbar sind. Dadurch fehlen einige Komponenten innheralb der Exportdatei, so dass nur ein begrenzter Anteil an Informationen in die Exportdatei übernommen werden kann.

11 Umfangs- und Flächenberechnung

Es kann nicht nur das Alter des Baumes, mittels Zählung der Jahrringe, ermittelt werden, sondern weitaus mehr Informationen aus den Jahrringen entnommen werden, wie zum Beispiel Klima, Witterung und Wachstum. Die Analyse des Wachstums einzelner Jahrringe ist Kernpunkt der Untersuchungen. Hier werden Umfang und Fläche jedes einzelnen Ringes ermittelt, um anhand dieser Informationen Aussagen über Umweltbedingungen beim Wachstum des Baumes in bestimmten Jahren nachweisen zu können. Die Analyse der Jahrringe auf Klima und Witterung ist nicht Bestandteil des Projektes und wird somit nicht näher betrachtet. Die Art dieser Analyse sehr kompliziert und schwer realisierbar, da die Stammscheiben in einer sehr hohen Qualität und vorpräpariert sein müssen, um möglichst genaue Informationen ermitteln zu können.

11.1 Ermittlung der Fläche eines Jahrringes

Die Ermittlung der Fläche der einzelnen Jahrringe erfolgt mittels eines Füllalgorithmus, der die Anzahl der Pixel für jede durch die Splines erzeugte Fläche ermittelt. Ein Pixel hat eine feste Größe, wobei die Seitenlänge eines Pixels 0.035 cm beträgt. Daraus folgt, dass die Fläche eines Pixels $0,001225\ cm^2$ beträgt. Dieser Wert ist eine Default-Einstellung des Systems und kann von Hand verändert werden. (Siehe Skalierbarkeit) Dieser Multiplikator wird auf die Anzahl der Pixel der jeweiligen Flächen angewendet, um den Flächeninhalt zu ermitteln. Der verwendete Füllalgorithmus ist der $BoundaryFill - Algorithmus$.

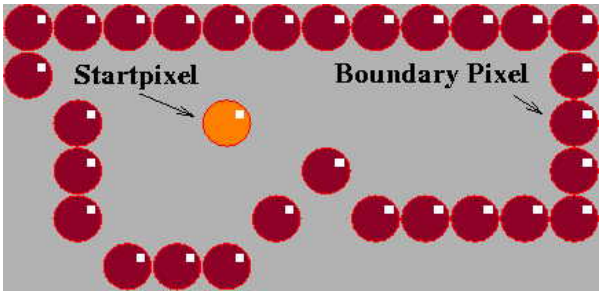

Abbildung 10: Boundary Fill - Algorithmus

Quelle: [Bil]

Bei diesem Algorithmus muss ein Pixel der zu füllenden Fläche bekannt sein. Ist der Pixel bekannt und befindet er sich im Innern der Fläche, kann der Algorithmus ausgeführt werden. Dieser Algorithmus kann rekursiv wie folgt beschrieben werden.

- Prüfen, ob es sich um ein Randpixel handelt (d.h. die Farbe des Pixels mit der Randfarbe übereinstimmt) oder ob der Pixel bereits gefüllt wurde (d.h. die Farbe des Pixels mit der Füllfarbe übereinstimmt)
- Falls nicht, wird der Pixel gefüllt und der Algorithmus rekursiv für die 4 bzw. 8 Nachbarpixel aufgerufen

```
/**
 * Input:
 * x, y             die x bzw. y Koordinate des Anfangspixels
 * fillColor        die Füllfarbe
 * boundaryColor    die Randfarbe
```

```
 */

function boundaryFill(x, y, fillColor, boundaryColor) {

    CurrentColor = getColorOfPixel(x, y);

    if(fillColor != currentColor AND boundaryColor != currentColor) {
        setPixel(x, y, fillColor);
        boundaryFill(x+1, y, fillColor, boundaryColor);
        boundaryFill(x-1, y, fillColor, boundaryColor);
        boundaryFill(x, y+1, fillColor, boundaryColor);
        boundaryFill(x, y-1, fillColor, boundaryColor);
    }
}
```

Quelle: [Bou06]

Der Vorteil dieses Algorithmus ist, dass er unabhängig von der geometrischen Form des zu füllenden Objektes verwendet werden kann, der Nachteil ist, dass die Rekursionstiefe sehr groß werden kann. (Bei zu hoher Rekursionstiefe treten Stack-Overflows auf) Alternativ lässt sich zum rekursiven Algorithmus ein iterativer Algorithmus erzeugen. Im Programm selbst wird die iterative Version des Algorithmus benutzt. Auf diese Art und Weise werden alle Bildpunkte einer zu füllenden Fläche ermittelt und dadurch kann die durch die Splines erzeugte Fläche ermittelt werden.

11.2 Iterative Version des Boundary Fill Algorithmus

Bei der iterativen Version des Boundary Fill Algorithmus gibt es einen Startvektor, der initial mit einem Element aus der zu füllenden Fläche startet. Jedes Element, welches dem Vektor hinzugefügt wird, enthält eine boolsche Variable. Diese Variable *hasPassed* gibt darüber Auskunft, ob ein Punkt schon einmal im Laufe der Schleifendurchläufe besucht worden ist oder nicht. Der Algorithmus ist in eine do-while Scheife gepackt, die beendet wird, wenn alle Punkte als *hasPassed* markiert wurden. Bis dies passiert, wird immer wieder der aktuelle Pixel im Vektor geholt, geprüft, ob es ein Randpunkt oder ein bereits besuchter Punkt ist und der Punkt als *hasPassed* markiert. Danach werden seine 4 anliegenden Punkte hinzugefügt und die Schleife geht an der Stelle weiter, wo sie aufgehört hatte.

```
do{
    // Im Vektor v werden alle besuchten Punkte gespeichert
    // (Initial ist in diesem Vektor ein Punkt aus der zu
    // füllenden Fläche enthalten)
    a = v.size();

    // Prüfe, ob alle Punkte besucht worden sind
    for (int j=0; j<a; j++){
        if(v.elementAt(j).hasPassed==true)
        b++;
    }

    // Wenn alle Punkte als passiert markiert wurden -> Fertig
    if (b==a){
        done = true;
    }

    for (int i=Lauf; i<a; i++){
        // aktuelle Punkt
        int x = v.elementAt(i).x;
        int y = v.elementAt(i).y;
        Lauf++;

        // Holen der Farbinformationen des Pixels
        PointColor = ip.getPixel(x, y, PointColor);

        // Ist der Punkt ein Randpunkt?
        if (PointColor==BoundaryColor){
            isEdge=true;
        }

        // Ist der Punkt schon einmal besucht worden?
        if (FillColor==PointColor){
            alreadyVisited = true;
        }

        // Prüfung ob zu hinzufügende Punkte schon besucht wurden
        // bzw. ein Randpunkt ist
        if (!isEdge && !alreadyVisited){
```

```
            // Füge Nachbarpunkte hinzu, wenn diese noch nicht besucht
            // worden sind
            if(!isVisited(x+1, y))
            v.addElement(new Point(x+1, y));  // rechts

            if(!isVisited(x-1, y))
                v.addElement(new Point(x-1, y));  // links

            if(!isVisited(x, y+1)){
                v.addElement(new Point(x, y+1)); // oben

            if(!isVisited(x, y-1))
                v.addElement(new Point(x, y-1)); // unten

            // zeichne aktuelles Pixel
            ip.putPixel(x, y, FillColor);
        }
        isEdge = false;
        alreadyVisited = false;
    }
    b=0;
}while(!done);
```

11.3 Ermittlung des Umfangs der Fläche

Die Berechnung des Umfangs kann abgleitet werden aus der Berechnung des Fläche. Beim Berechnen der Flächen wird die Anzahl der berührten Randpunkte gespeichert. Berührte Randpunkte werden als markiert gekennzeichnet, so dass kein Punkt doppelt vorkommen kann. Dadurch erhält man näherungsweise die Anzahl der Pixel, die die Fläche umspannen. Beim ersten Ring, das ist der kleinste Ring, haben wir den reinen Umfang des Ringes. Beim zweiten Ring erhalten wir den Umfang des zweiten Ringes und des ersten Ringes. Jetzt wird der Umfang des ersten Ringes vom Umfang des zweiten Ringes subtrahiert, um den richtigen Umfang des zweiten Ringes zu erhalten. Dieses Verfahren wird bis zum letzten zu füllenden Ring fortgeführt, um die jeweiligen Umfänge der Ringe zu ermitteln.

12 Zusammenfassung und Ausblick

Bei der Entwicklung der Anwendung traten mehrere interessante Erkenntnisse zu Tage. Die automatisierte Bildanalyse ist in sehr vielen Bereichen auf dem Vormarsch und erleichtert vielen die Arbeit. Allerdings birgt sie auch Gefahren, gerade in der Medizin, wenn man sich stur auf die automatisierte Analyse von Problemen verlässt. Sie dient lediglich dazu, Hinweise zu liefern und Ansätze für Lösungen zu präsentieren, nicht die Lösung selbst. Für die Problematik, mit der ich mich beschäftigt habe, ist eine automatisierte Bildanalyse sehr unangebracht, da es sehr viele Randbedingungen gibt, die eine Analyse erschweren bzw. unmöglich machen. Deswegen habe ich mich für eine semiautomatisierte Bildanalyse entschieden. Der Vorteil liegt daran, dass der Benutzer von Hand die zu untersuchenden Informationen eingeben kann, eine Lösung berechnen kann und diese dann überprüfen kann auf Genauigkeit und Effizienz. Die Analyseverfahren, die benutzt wurden, haben alle unterschiedliche Stärken und Schwächen, die im Schnitt alle Problemfelder abdecken, aber alleine einige Schwachstellen aufweisen, wie z.B. die Oszillationen bei sehr enger Punktekonstellation. Hier wäre natürlich von Vorteil, beide Verfahren mathematisch zu kombinieren, um die Schwachstellen des einen Verfahrens durch die Stärken des anderen Verfahrens auszugleichen. Ich habe leider kein Verfahren gefunden, welches genau auf meinen Zweck der Arbeit ausgelegt gewesen ist. Auch ist es mir nicht gelungen, beide Verfahren mathematisch derart miteinander zu verknüpfen, um die Vorteile beider Verfahren zu kombinieren. Die Verfahren, die ich benutze zur Analyse, sind kubische Splines und Parabolic Blending. Die Ergebnisse beider Verfahren sind im Prinzip sehr ähnlich und unterscheiden sich nur in mehr oder weniger kleinen Details. Die Unterschiede sind grafisch in Abb. 8 und Abb. 9 genauer dargestellt.
Das Programm/Plugin ist auf jeden Fall eine Erleichterung der Analyse von Stammscheiben im Gegensatz zu der bisherigen Art und Weise, wie diese analysiert wurden. Das Verfahren, welches bisher am Institut für Forstliche Biometrie und Informatik der Universität Göttingen benutzt worden ist, ist sehr kompliziert und zeitaufwändig. Es ist mit mathematischen Mitteln geschafft worden, komplizierte und manuelle Prozesse zu vereinfachen, indem Algorithmen anhand von wenigen interaktiv eingegebenen Bildinformationen erfolgreich Jahresringe verfolgen können. Die Genauigkeit der Algorithmen ist von Stammscheibe zu Stammscheibe unterschiedlich und hängt von deren Qualität ab. Des weiteren verfügt das Programm über eine Exportfunktion. Diese Exportfunktion kann allerdings nicht alle Informationen für das in Göttingen befindliche Analysesystem reproduzieren, weil wichtige Informationen, um einen erfolgreichen Import zu gewährleisten, nur durch das in Göttingen befindliche Analysesystem gegeben sind. Es wurden auch keine Tests durchgeführt, um die Exportdatei auf Funktionstüchtigkeit im Göttinger Analysesystem zu testen, so dass die aktuelle Exportdatei, zwar alle, während der Analyse mit dem Plugin, ermittelten Daten zur Verfügung stellt, aber es nicht gewährleistet werden kann, ob das System in Göttingen diese erfolgreich importieren kann.

13 Anhang - Bedienungsanleitung

13.1 Öffnen der Datei

Zum Öffnen des zu bearbeitenden Bildes gibt es zwei Möglichkeiten. Die Erste ist die Tastenkombination STRG+O und die Zweite ist über die Menüleiste File → Open, um den Dialog zum Öffnen der Datei anzuzeigen.

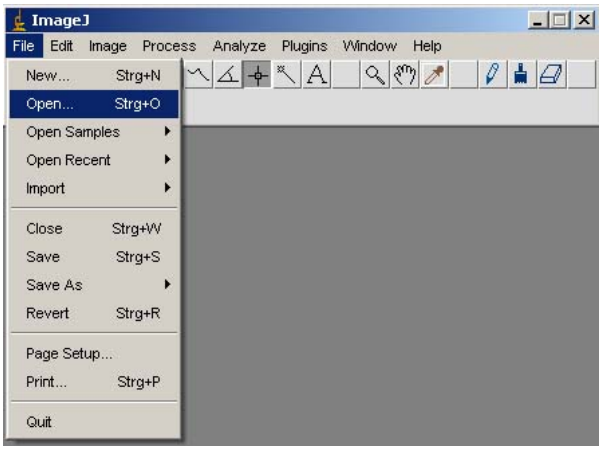

Abbildung 11: Öffnen des Bildes

13.2 Auswahl des Mittelpunktes

Noch bevor das eigentliche Plugin gestartet werden kann, muss der Mittelpunkt festgelegt werden. Dazu wird im Selektionsmenu 'Point Selektion' (siehe Abbildung 12) aktiviert. Nach dem Aktivieren dieser Selektionsart muss mit der Tastenkombination 'SHIFT + Linksklick' der Mittelpunkt im Bild gesetzt werden. Es können zwar mehrere Punkte gesetzt werden, allerdings wird nur der erste Punkt als Mittelpunkt akzeptiert.

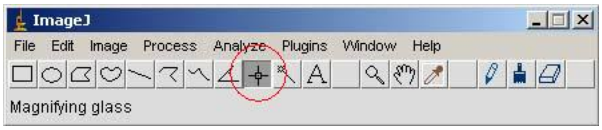

Abbildung 12: Auswahl des Mittelpunktes

13.3 Starten des Plugins

Vorraussetzung dafür, dass das Plugin erfolgreich gestartet werden kann, ist, dass das zu bearbeitende Bild geladen und der Mittelpunkt gesetzt sein muss. Danach wird über die

Menüleiste das Plugin wie folgt gestartet: Plugins → Name des Plugin Verzeichnisses → Name des Plugins. (siehe Abbildung 13)

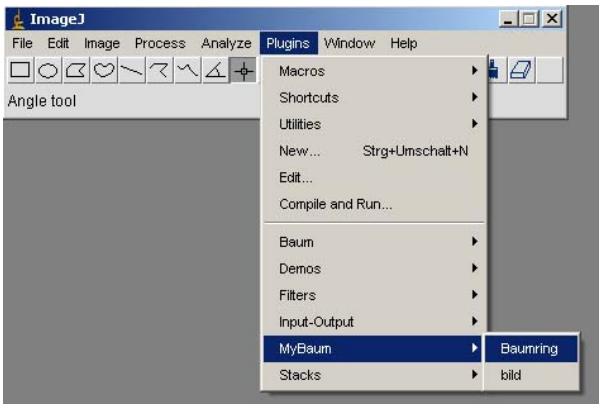

Abbildung 13: Starten des Plugin

13.4 Sprachauswahl

Nach dem das Plugin gestartet worden ist, erscheint ein Dialogfenster. Der Benutzer hat jetzt die Möglichkeit zu einer Sprachauswahl. Es stehen ihm zwei Sprachversionen zur Auswahl, Englisch und Deutsch. Je nachdem für welche Sprache sich der Benutzer entscheidet, werden alle weitere Interaktionsdialoge in der ausgewählten Sprache angezeigt. (siehe Abbildung 14) In der weiterführenden Dokumentation wird mit dem englischen Sprachpaket gearbeitet. An Stellen, wo englische Aktionen ausgeführt werden, wird zusätzlich der deutsche Name der Aktion aufgeführt.

Abbildung 14: Sprachauswahl

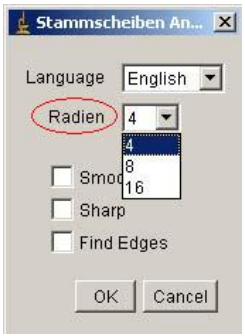

Abbildung 15: Radienauswahl

13.5 Radienauswahl

Bei der Radienauswahl wird das Bild in mehrere Segmente durch sichtbare Linien unterteilt. Diese Linien dienen der Wahrung der Übersichtlichkeit. Zur Auswahl stehen 4, 8 und 16. Bei kleinen Bildern ist es ratsam das Bild in 4-8 Radien zu unterteilen und bei größeren Bildern 16 Radien auszuwählen. (siehe Abbildung 15)

13.6 Glätten/Weichzeichner

Mit diesem Filter besteht die Möglichkeit etwas Schärfe aus dem Bild zu nehmen, das Gegenteil zum Schärfe Filter. Um diese Option zu benutzen muss der Haken im Feld *Smooth* gemacht werden. (siehe Abbildung 16)

13.7 Schärfen

Das Schärfen des Bildes soll es dem Benutzer ermöglichen qualitativ nicht hochwertige Bilder etwas aufzuwerten in deren Qualität. Um diese Option zu benutzen muss der Haken im Feld *Sharp* gemacht werden. (siehe Abbildung 17)

13.8 Automatische Kantenerkennung

Bei dieser Option wird auf das Originalbild ein Kantenerkennungsalgorithmus angewendet, in diesem Fall der so genannte Sobel - Operator. Dieser Algorithmus versucht automatisch Unterschiede in Farbverläufen zu erkennen und Kanten hervorzuheben.

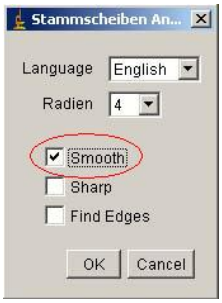

Abbildung 16: Smooth

Abbildung 17: Sharp

Abbildung 18: Find Edges

Um diese Option zu benutzen muss der Haken im Feld 'Find Edges' gemacht werden. (siehe Abbildung 18)

13.9 Festlegen der Stützpunkte

Das Festlegen der Stützpunkte funktioniert nach dem gleichen Prinzip, wie das Setzen des Mittelpunktes. Im Selektionsmenü muss 'Point Selection' aktiviert sein (sie-

he Abbildung 12). Danach werden die Punkte im Bild mittels der Tastenkombination SHIFT+Linksklick im Bild platziert. Es können auf diese Art und Weise mehrere Punkte im Bild platziert werden, die dann zur Berechnnung des Splines einbezogen werden.

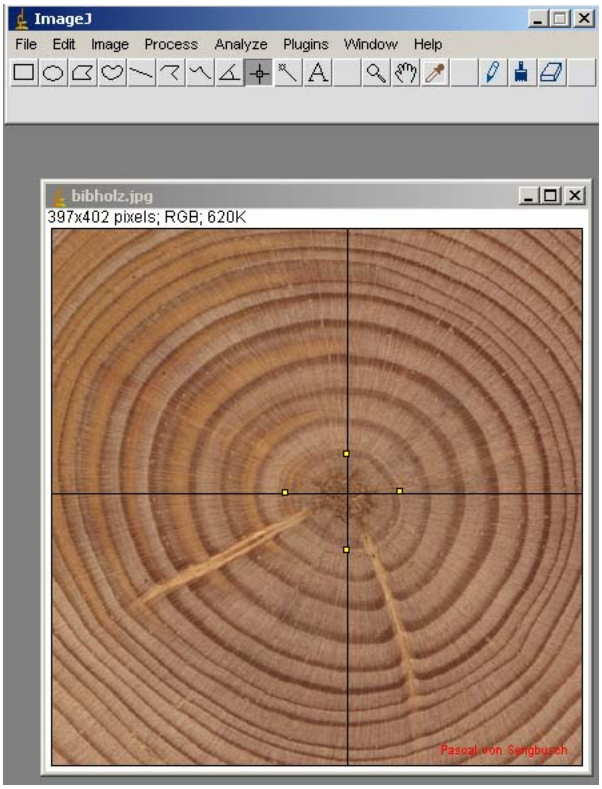

Abbildung 19: Festlegen der Stützstellen

13.10 Auswahl der Splineinterpolationsalgorithmen

Die Option 'Choose your method' ist für die Auswahl des Splineinterpolationsalgorithmus zuständig. Hier stehen 3 Möglichkeiten zur Verfügung. Die erste Möglichkeit ist die Erzeugung 'geschlossener, kubischer Splines' und die zweite Möglichkeit ist das 'Parabolic Blending' Verfahren. Bei der dritten Möglichkeit wird nach beiden Verfahren der Spline berechnet und beide Ergebnisse bzw. Splines im Bild dargestellt.

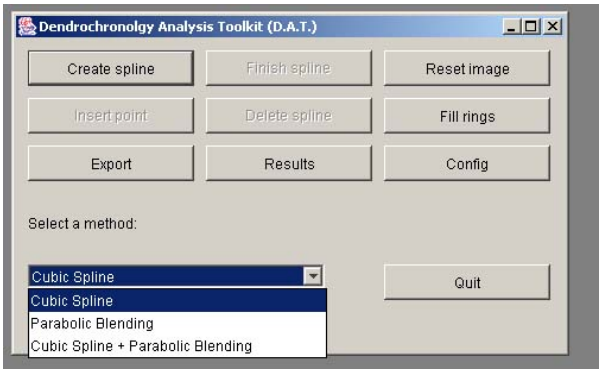

Abbildung 20: Auswahl der Splineinterpolationsalgorithmen

13.11 Erzeugen der Splines

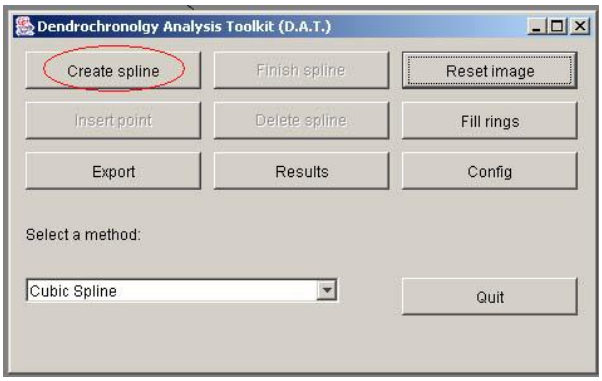

Abbildung 21: Erzeugen des Splines

Bevor der Spline erzeugt bzw. berechnet werden kann, müssen der Mittelpunkt gesetzt, die Stützpunkte festgelegt (mindestens 4 Stützpunkte) und ein Algorithmus ausgewählt worden sein. Sind alle drei Bedingungen erfüllt, kann 'Create Spline' bzw. 'Erzeuge Spline' (siehe Abbildung 21) betätigt werden und es wird der Spline erzeugt bzw. berechnet. (siehe Abbildung 22)

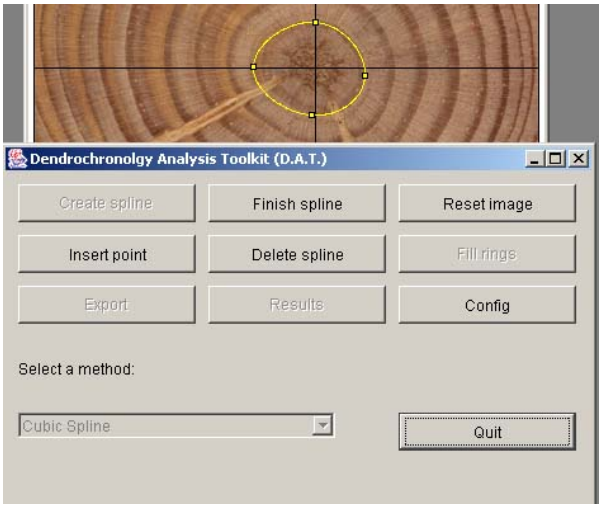

Abbildung 22: Erzeugter Spline

13.12 Punkte hinzufügen

Punkte hinzufügen funktioniert fast analog wie die Vorgehensweise der Erzeugung des Splines. Die zusätzlichen Punkte werden an den nötigen Stellen mittels SHIT+Linksklick hinzugefügt. Nun wird 'Insert Point' bzw. 'Punkt hinzufügen' gedrückt (siehe Abbildung 23), der Spline wird neu berechnet und neu dargestellt.

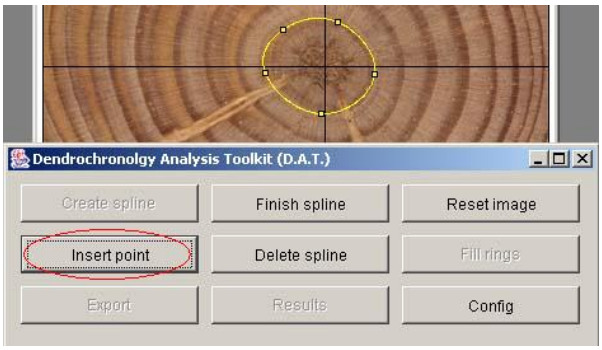

Abbildung 23: Stützstellen hinzufügen

13.13 Spline entfernen

Es besteht auch die Möglichkeit den kompletten Spline zu entfernen. Dazu muss 'Delete' bzw. 'Entfernen' gedrückt werden. Diese Option entfernt NUR den aktuellen Spline,

falls der Benutzer z.B. Stützstellen entfernen will oder einen anderen Splinealgorithmus anwenden will.

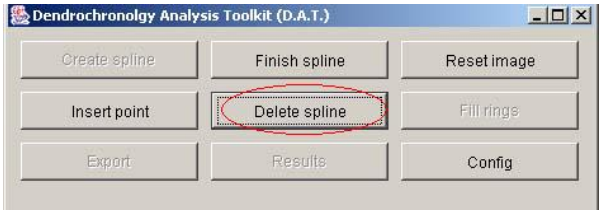

Abbildung 24: Spline entfernen

13.14 Spline abschliessen

Wenn Sie der Meinung sind, dass der Spline fertig gestellt ist, müssen Sie den Spline abschliessen. Dazu muss 'Finish spline' bzw. 'Spline abschliessen' betätigt werden. Danach können Sie wieder neue Splines erstellen. Zusätzlich kommt eine Sicherheitsabfrage, ob diese Aktion wirklich durchgeführt werden soll.

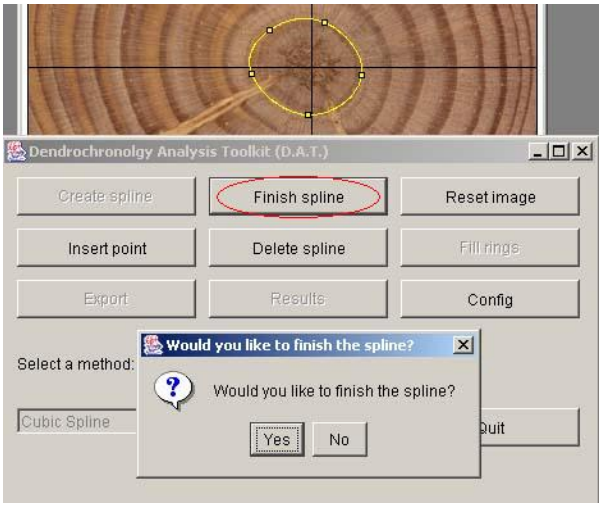

Abbildung 25: Spline abschliessen

Ein besonderer Fall tritt auf, wenn beide Verfahren ausgewählt wurden. Beim Abschliessen des Spline bekommt der Benutzer einen Dialog zur Auswahl, indem er ein der beiden Verfahren auswählen muss. (siehe Abb. 26)

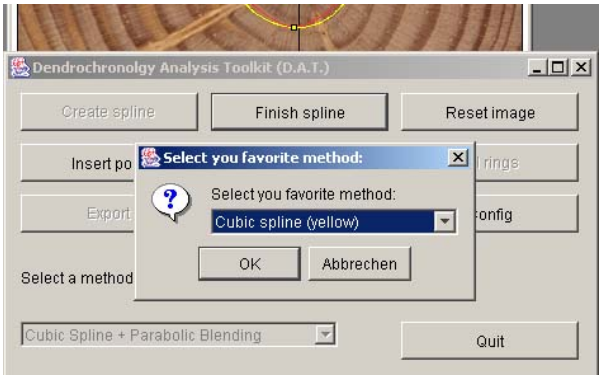

Abbildung 26: Spline abschliessen - beide Verfahren ausgewählt

13.15 Punkte verschieben und Spline neu berechnen

Falls Punkte zufällig an der falschen Position gesetzt worden sind, besteht die Möglichkeit die Position der Punkte zu korrigieren. Dazu wird mit dem Mauszeiger auf den zu verschiebenden bzw. zu korrigierenden Punkt gegangen bis das Icon des Zeigefinger erscheint. Wenn dieses Icon erschienen ist, kann der Punkt an die gewünschte Position gerückt bzw. verschoben werden. Nach dem die Position der Punkte korrigiert worden ist, muss 'Insert point' bzw. 'Punkt hinzufügen' geklickt werden. (siehe Abbildung 23)

13.16 Bild zurücksetzen

Diese Option setzt alle erzeugten Bildinformationen zurück und stellt das Originalbild wieder her. Auch hier erscheint eine Sicherheitsabfrage, ob die ausgewählte Aktion wirklich durchgeführt werden soll.

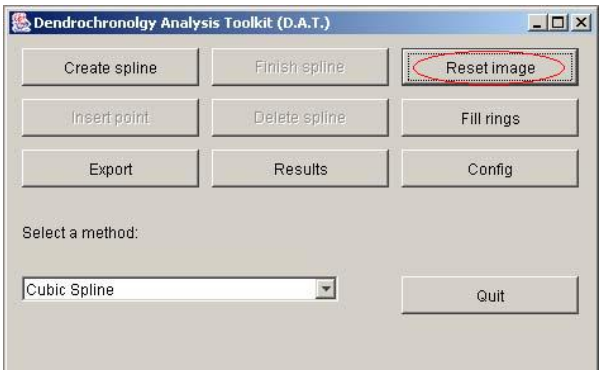

Abbildung 27: Bild zurücksetzen

13.17 Exportfunktion

Die Exportfunktion ist dafür zuständig, dass die gesammelten Informationen aufbereitet in ein vorgeschriebenes Format gebracht werden, exportiert und in einem in Göttingen befindlichen System importiert werden können. Um diese Möglichkeit zu nutzen, muss der 'Export' Button gedrückt werden. (siehe Abbildung 28)

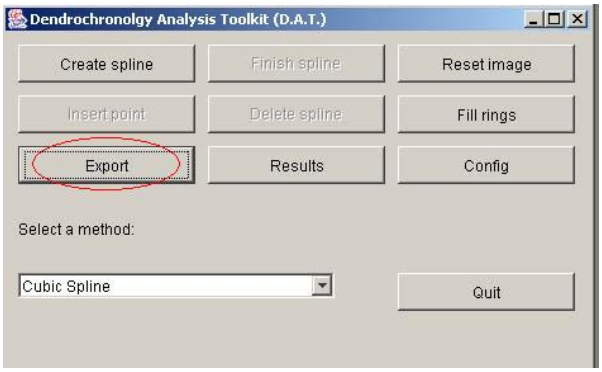

Abbildung 28: Export

13.18 Skalierbarkeit

Falls Bilder nicht im Maßstab 1 : 1 sind besteht die Möglichkeit, das Bild zu kalibrieren. Dazu muss als erstes *Straight line selection* ausgewählt werden. (siehe Abb. 29)

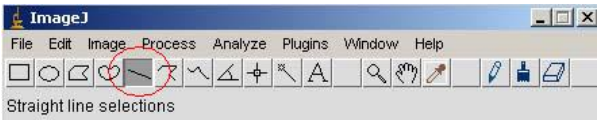

Abbildung 29: Straight Line Selection

Danach muss eine vertikale oder horizontale Linie in das Bild gezeichnet werden. (in Abb. 30 am linken Bildrand) Es ist besonders wichtig, dass diese Linie horizontal oder vertikal ist, da dadurch der Faktor der Skalierung genau berechnet werden kann. Entspricht die Linie nicht der Vorgabe, erscheint eine Fehlermeldung. Nachdem die Linie gezeichnet worden ist, muss *Config* geklickt werden, um das Dialogfenster zu öffnen, in das der Skalierungswert eingegeben werden muss. (siehe Abb. 30)

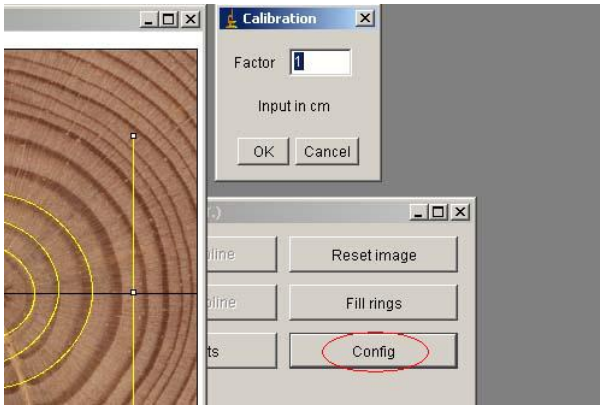

Abbildung 30: Kalibrierung

Anschliessend erfolgt eine Bestätigung der gemachten Änderungen. (siehe Abb. 31)

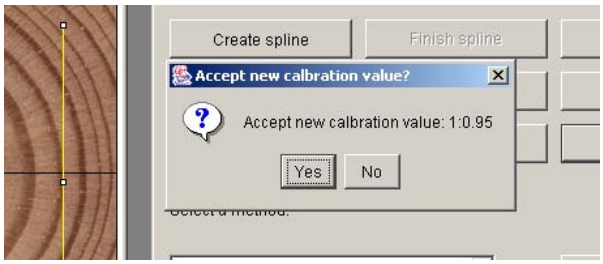

Abbildung 31: Bestätigen der Änderungen

13.19 Ringe darstellen

Um die Fläche und den Umfang eines Jahrringes ermitteln zu können, müssen vorher die Ringe gefüllt werden. Durch den Algorithmus, welcher die Fläche füllt, kann dann näherungsweise die Fläche und der Umfang für jeden Ring ermittelt werden. Die grafische Darstellung des Füllens ist Bestandteil des Verfahrens zur Berechnung der Fläche und des Umfanges und kann rückgängig gemacht werden (durch erneutes Klicken auf den Button). Nachdem die Ringe gefüllt worden sind, kann auf Ergebnisse(Results) geklickt werden, um sich die ermittelten Flächen und Umfänge anzeigen zu lassen (siehe Abb. 34).

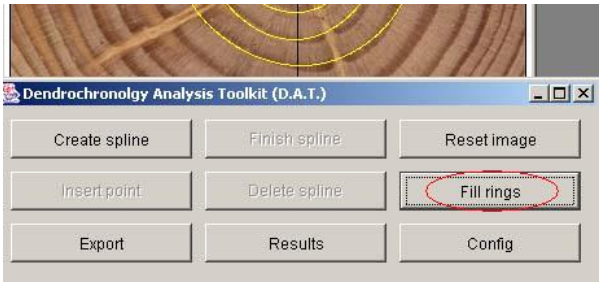

Abbildung 32: Füllen der durch die Splines erzeugten Flächen

In der Ergebnisausgabe wird die Fläche in cm^2 und der Umfang in cm angegeben.

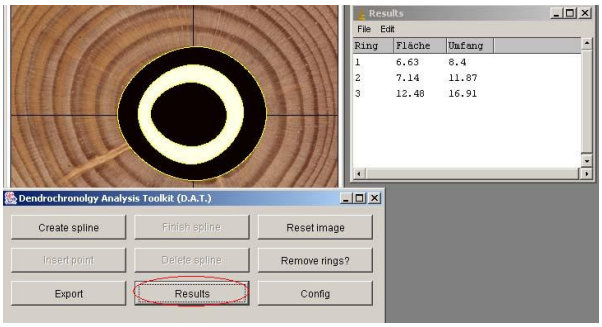

Abbildung 33: Anzeigen der Messergebnisse

Inhaltsverzeichnis

1 Einleitung **2**

2 Motivation **2**

3 Der Baumring **3**

4 Morphologische Bildbearbeitung **4**
4.1 Bildfilterung 4
4.2 Bildsegmentierung 4
4.3 Bildmessungen 4

5 Kantenerkennung **5**

6 Splines **5**
6.1 Kubische Splines 6
6.2 B-Splines 12

7 Parabolic Blending **13**

8 Umsetzung **15**

9 Vergleich der Verfahren **17**

10 Die Exportfunktion **19**

11 Umfangs- und Flächenberechnung **20**
11.1 Ermittlung der Fläche eines Jahrringes 21
11.2 Iterative Version des Boundary Fill Algorithmus 22
11.3 Ermittlung des Umfangs der Fläche 24

12 Zusammenfassung und Ausblick **25**

13 Anhang - Bedienungsanleitung **26**
13.1 Öffnen der Datei 26
13.2 Auswahl des Mittelpunktes 26
13.3 Starten des Plugins 26
13.4 Sprachauswahl 27
13.5 Radienauswahl 28
13.6 Glätten/Weichzeichner 28
13.7 Schärfen 28
13.8 Automatische Kantenerkennung 28
13.9 Festlegen der Stützpunkte 28

13.10Auswahl der Splineinterpolationsalgorithmen 29
13.11Erzeugen der Splines 30
13.12Punkte hinzufügen 31
13.13Spline entfernen 31
13.14Spline abschliessen 32
13.15Punkte verschieben und Spline neu berechnen 33
13.16Bild zurücksetzen 33
13.17Exportfunktion 34
13.18Skalierbarkeit 34
13.19Ringe darstellen 35

Literatur

[bau06] *Baum*. http://de.wikipedia.org/wiki/Baum. 2006. – Zugriffsdatum: 2006-31-07 08:02

[Bil] *Boundary Fill Algorithmus, grafische Darstellung*. http://wwwcs.uni-paderborn.de/cs/ag-domik/computergrafik/cg_skript/html/node51.htm. – Zugriffsdatum: 2006-31-09 08:02

[BK98] Bässmann, Henning ; Kreyss, Jutta: *Bildverarbeitung - Ad Oculus*. Springer Verlag Berlin Heidelberg, 1998

[Bou06] *Boundary-Fill-Rekursiv*. http://www.iam.unibe.ch/~fcglib/special_www/cg_lecture/lectContent/polygon_filling/lessons/boundaryfill/boundaryfill_theory.php. 2006. – Zugriffsdatum: 2006-31-09 08:02

[con] *Constrained Cubic Spline Interpolation*. http://www.korf.co.uk/spline.pdf. – Zugriffsdatum: 2006-31-07 08:02

[cub06] *Cubic Spline*. http://mathworld.wolfram.com/CubicSpline.html. 2006. – Zugriffsdatum: 2006-31-07 08:02

[cur06] *Curve Fitting*. http://kr.cs.ait.ac.th/~radok/math/mat7/step28.htm. 2006. – Zugriffsdatum: 2006-31-07 08:02

[den06] *Dendrochronologie*. http://de.wikipedia.org/wiki/Dendrochronologie. 2006. – Zugriffsdatum: 2006-31-07 08:02

[jah06] *Jahresring*. http://de.wikipedia.org/wiki/Jahresring. 2006. – Zugriffsdatum: 2006-31-07 08:02

[kub] *Kubische Splines*. http://www.arndt-bruenner.de/mathe/scripts/kubspline.htm. – Zugriffsdatum: 2006-31-07 08:02

[kub00] *Kubische Spline Interpolation*. http://www.we.fh-osnabrueck.de/fbwe/vorlesung/edv2/octave/node15.html. 2000. – Zugriffsdatum: 2006-31-07 08:02

[Kur04] Kurth, Prof. Dr. W.: *Computergrafik*. http://www-gs.informatik.tu-cottbus.de/~wwwgs/cg2_v09a.pdf. 2004. – Zugriffsdatum: 2006-31-07 08:02'

[Soi98] Soille, Pierre: *Morphologische Bildbearbeitung*. Springer Verlag Berlin Heidelberg, 1998

[spla] *Spline interpolation*. http://en.wikipedia.org/wiki/Natural_cubic_spline. – Zugriffsdatum: 2006-31-07 08:02

[splb] *Splines*. http://www-lehre.informatik.uni-osnabrueck.de/~cg/2000/skript/7_2_Splines.html. – Zugriffsdatum: 2006-31-07 08:02

[spl06] *Spline*. http://de.wikipedia.org/wiki/Spline. 2006. – Zugriffsdatum: 2006-31-07 08:02

Abbildungsverzeichnis

1 Stammscheibe . . . 3
2 Kantenhervorhebung . . . 4
3 durch den Algorithmus erzeugter natürlicher Spline . . . 12
4 B-Spline mit n=4 . . . 12
5 Parabolic Blending . . . 13
6 Einen Jahrring interpolierende Splinekurve . . . 16
7 Oszillationen . . . 17
8 kubischer Spline . . . 19
9 Parabolic Blending . . . 19
10 Boundary Fill - Algorithmus . . . 21
11 Öffnen des Bildes . . . 26
12 Auswahl des Mittelpunktes . . . 26
13 Starten des Plugin . . . 27
14 Sprachauswahl . . . 27
15 Radienauswahl . . . 27
16 Smooth . . . 28
17 Sharp . . . 28
18 Find Edges . . . 28
19 Festlegen der Stützstellen . . . 29
20 Auswahl der Splineinterpolationsalgorithmen . . . 30
21 Erzeugen des Splines . . . 30
22 Erzeugter Spline . . . 31
23 Stützstellen hinzufügen . . . 31
24 Spline entfernen . . . 32
25 Spline abschliessen . . . 32
26 Spline abschliessen - beide Verfahren ausgewählt . . . 33
27 Bild zurücksetzen . . . 33
28 Export . . . 34
29 Straight Line Selection . . . 34
30 Kalibrierung . . . 35
31 Bestätigen der Änderungen . . . 35
32 Füllen der durch die Splines erzeugten Flächen . . . 36
33 Anzeigen der Messergebnisse . . . 36