UNIVERSITÄT SALZBURG

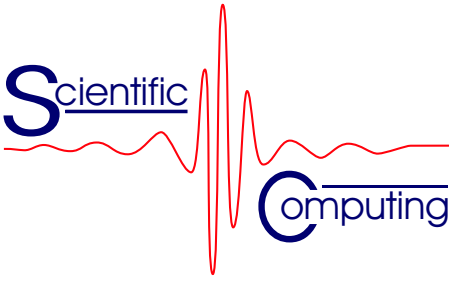
Algorithmenbibliothek für PSLG-Grafikeditoren mit effizienter Online-Validierung zum Einsatz in Tunneldokumentationssystemen

Gerhard Mitterlechner[ab] Martin Held[a] Franz Gusenbauer[b]

[a]Department of Scientific Computing
[b]Fa. Geo-Byte GmbH, A-5020 Salzburg

Technical Report 2005-05 August 2005

Department of Scientific Computing



Jakob-Haringer-Straße 2
5020 Salzburg
Austria
www.scicomp.sbg.ac.at

Technical Report Series

Algorithmenbibliothek für PSLG-Grafikeditoren mit effizienter Online-Validierung zum Einsatz in Tunneldokumentationssystemen

Gerhard MITTERLECHNER, Martin HELD und Franz GUSENBAUER

Zusammenfassung

Zentraler Bestandteil eines Tunneldokumentationssystems ist der Grafikeditor zur digitalen Eingabe von Tunnelquerschnittsdaten. Um eine benutzerfreundliche, rasche und vor allem konsistente Modellierung der geometrischen Struktur eines Tunnelquerschnitts zu ermöglichen, ist ein Editor nötig, welcher Funktionalität zum interaktiven Erstellen und Bearbeiten von polygonal begrenzten Gesteinsflächen mit beliebiger Schachtelungstiefe einerseits in visueller und andererseits in algorithmen- und datenstrukturorientierter Hinsicht bereitstellt. Hierbei entspricht je ein PSLG (*Planar Straight Line Graph*) der Geometrie des gesamten Tunnelquerschnitts („Ortsbrust“).
Diese Arbeit stellt die wichtigsten methodischen Ansätze einer anwendungsgebiet- und GUI-unabhängigen Implementierung der grundlegenden Datenstrukturen und Algorithmen für notwendige Editier-Operationen auf PSLGs vor. Auf Grund der weiten Verbreitung von PSLGs zur Repräsentierung der Geometrie von geographischen oder geologischen Objekten ist das Einsatzgebiet unserer Editor-Bibliothek nicht nur auf die Modellierung von Tunnelquerschnitten oder Gesteinsflächen beschränkt, sondern sie ist auch für andere GIS-Anwendungen einsetzbar, bei denen die Geometrien PSLGs entsprechen.

1 Einleitung

Bei der Errichtung von Tunnelbauwerken liegt das Hauptaugenmerk auf langer Lebensdauer, höchst möglicher Sicherheit und niedrigen Baukosten. Die Österreichische Tunnelbauweise ist eine international anerkannte Methode, Baugrund (Gebirge) und Bauwerk als Einheit zu sehen und in Abhängigkeit vom Baugrund Ausbaumaßnahmen festzulegen, die einerseits die kostengünstige Errichtung des Bauwerkes erlauben, andererseits aber auch auf die Sicherheit des Bauwerkes Bedacht nehmen. Darum wird im Zuge ausführlicher Erkundungsmaßnahmen versucht, ein möglichst genaues Bild der geologischen Verhältnisse zu erhalten. Entsprechend den geologischen Verhältnissen wird das zu durchquerende Gebirge in Bereiche mit ähnlichem Ausbruchverhalten eingeteilt. Für jeden derartigen Homogenbereich werden die Ausbaumaßnahmen (wie Anzahl und Länge der Anker, Spritzbetonstärke, etc.) festgelegt.
Im Zuge der Errichtung von Tunnelbauwerken sind die angetroffenen geologischen Verhältnisse zu dokumentieren (ÖGG 2001). Dies hat einen doppelten Zweck – einerseits dient diese Dokumentation der Abrechnung der Errichtungskosten und andererseits ist eine genaue Dokumentation die Basis für die Kontrolle der Standsicherheiten des Tunnels. Tunneldokumentationssysteme unterstützen die Geologen bei der Dokumentationsarbeit.
Der typische Ablauf einer Tunneldokumentation ist folgendermaßen: Geologen fertigen zur Dokumentation der Geologie in regelmäßigen Abständen 2D-Handskizzen der Geologie der

Ortsbrust an, welche später mithilfe des Grafikeditors des Tunneldokumentationssystems visualisiert und digitalisiert wird. Der auf diese Weise aufgebaute 3D-Datenbestand ist Basis für verschiedene Ableitungen, etwa für statistische Auswertungen, Profilschnitte oder die Ableitung eines 3D-Modells (Abb. 1) der Tunnelgeologie.
Die geometrische Struktur einer Ortsbrust ist als ein mathematisches Objekt repräsentierbar, was Grafikeditoren, welche zur Tunneldokumentation eingesetzt werden, ausnützen können, um eine rasche, korrekte und konsistente Dateneingabe zu ermöglichen. Wie weiter unten detailliert erläutert, unterstützen traditionelle Grafikeditoren das Editieren dieser Objekte nur unzureichend, weshalb wir für die Ortsbrustvisualisierung und -digitalisierung einen auf diese spezielle Struktur hin ausgelegten Grafikeditor entwickelten. Derzeit wird ein auf unserer Bibliothek basierender Editor von der Fa. *Geo-Byte* im Zuge der Adaptierung der Tunneldokumentationsapplikation *2doc* eingesetzt (FURTMÜLLER&MARSCHALLINGER 2003).

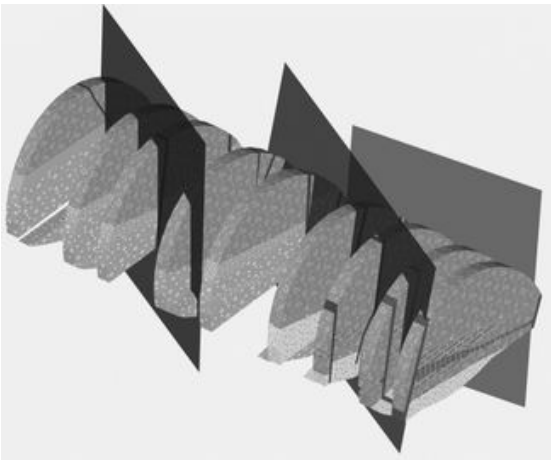


Abb. 1.: Ein aus 2D-Tunnelquerschnitten abgeleitetes 3D-Modell eines Tunnels.

Konkret wird die Ortsbrust als eine Menge von polygonalen Flächen dargestellt, welche von (selbst-)überschneidungsfreien (d. h. *einfachen*) Polygonketten berandet sind; siehe Abb. 2 und Abb. 3. Eine derartige Anordnung von Polygonketten wird in der Theorie[1] der Algorithmischen Geometrie als *Planar Straight Line Graph* (PSLG) bezeichnet. Ein PSLG ist also die graphische Repräsentierung eines planaren Graphen in der Ebene, wobei alle Kanten des Graphen Geradensegmente sind. Weitere bekannte GIS Objekte, welche als PSLGs repräsentiert werden können, sind beispielsweise Versorgungsnetzwerke, Straßen und Landkarten.
Die Gesteinsflächen werden durch vom Benutzer editierte Polygone repräsentiert, welche den Innengebieten des PSLG entsprechen. Der gesamte Tunnelquerschnitt wird durch die strukturierte Anordnung mehrerer derartiger Polygone in einem einzigen PSLG modelliert. Zu beachten ist der durchaus häufig auftretende Fall, dass eine Gesteinsschicht eine weitere

[1] Interessierte Leser finden unter http://student.cosy.sbg.ac.at/~gmitter/pslg-editor.html eine mathematisch-formale Festlegung der für PSLGs wichtigsten graphentheoretischen Terme. Weiters sind als Ergänzung zu dieser Publikation mehrere Farbbilder bereitgestellt.

Gesteinsschicht komplett umschließt, wie z. B. in Abb. 3. Geometrisch bedeutet dies, dass ein Polygon ein oder mehrere weitere Polygone als *Inseln* enthalten kann, welche jeweils ggf. wiederum Inseln enthalten können. Die Theorie spricht dann von beliebig *geschachtelten* Polygonen, sowie davon, dass der PSLG in mehrere *Zusammenhangskomponenten* (ZHK) zerfällt.

Abb. 2: Handskizze einer Ortsbrust

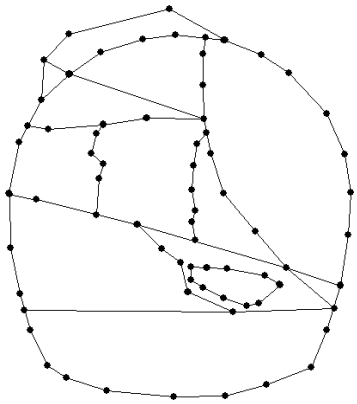


Abb. 3: PSLG der Ortsbrust

2 Anforderungen an einen Grafikeditor für PSLGs

Folgende Faktoren sind für einen allgemeinen PSLG-Editor von zentraler Bedeutung: a) die Online-Validierung von Benutzereingaben, um Inkonsistenzen bezüglich der geometrisch-topologischen Eigenschaften des PSLG zu vermeiden, b) die automatische Vervollständigung von Polygonketten zu neuen Gebieten des PSLG und deren korrekte Integration in den PSLG, c) die robuste und effiziente Implementierung der zugrundeliegenden geometrischen Algorithmen, um selbst bei großen Datensätzen akzeptable Laufzeiten zu garantieren. Speziell aus geologischer Sicht wichtig sind weiters a) ein standardisiertes generisches Datenformat wie XML für die Darstellung und letztlich Rekonstruktion der Geometrie und Topologie des Tunnelquerschnitts, b) ein auf Kernfunktionalität beschränktes User Interface (GUI), sowie ggf. Multi-User Fähigkeit.

Insbesondere muss ein PSLG-Grafikeditor Operationen bereitstellen, welche es erlauben, PSLGs den Benutzerwünschen entsprechend möglichst rasch und einfach zu erstellen und zu modifizieren. Das bedeutet speziell, dass der Benutzer analog zu bekannten Grafikeditoren mithilfe eines Eingabegeräts (z. B. Maus) neue Elemente (meist Polygone) dem PSLG hinzufügen, alte Elemente modifizieren (z. B. durch Teilung) und bestehende Elemente (z. B. Ecken) verschieben kann. Um möglichst benutzerfreundlich und rasch Editierungen vornehmen zu können, werden z. B. Polygonketten, die an ein bereits bestehendes Polygon angehängt werden, automatisch, d. h. ohne weiter erforderliche Benutzeraktionen, zu neuen Polygonen ergänzt, welche in den PSLG integriert werden. (Details dazu in Abschnitt 3.)

Derjenige Teil der Editor-Software, der für die Verwaltung des PSLG zuständig ist, muss allein aufgrund von Informationen des Eingabegeräts, also 2D-Koordinaten der jeweiligen Position des Maus-Cursor, für einen korrekten und konsistenten Aufbau sowie eine geeignete Repräsentierung des PSLG im Speicher sorgen. Die algorithmische Hauptschwierigkeit

besteht dabei darin, auf Basis von geometrischen Informationen Aussagen über die Topologie des PSLG zu treffen, wobei sich hier der Begriff Geometrie konkret auf die 2D-Koordinaten der Ecken des PSLG und der Begriff Topologie auf Wissen über die Gesamtstruktur des PSLG bezieht (wie z. B. hierarchische Beziehungen betreffend die Schachtelung von Polygonen). Die im Editierprozess entstandenen geometrischen Veränderungen bewirken Veränderungen in der Topologie des PSLG. Sie dürfen nur derart vorgenommen werden, dass die geometrische und topologische Korrektheit und Konsistenz der Ortsbrust gewahrt bleiben. Sämtliche PSLG-Eigenschaften müssen eingehalten werden, insbesondere darf eine geometrisch-topologische Eigenschaft des PSLG nicht im Widerspruch zu einer anderen stehen. (Z. B. muss der Schnittpunkt zweier Geraden auch tatsächlich als auf beiden Geraden liegend erkannt werden.) Anders ausgedrückt: jede Benutzeroperation muss vor ihrer tatsächlichen Ausführung, d. h. vor der tatsächlichen Modifikation des im Speicher befindlichen PSLG, hinsichtlich Geometrie und Topologie *validiert* werden. Wird der Benutzer dabei bei jeder Cursor-Bewegung informiert, ob das Ausführen einer Modifikation an der aktuellen Cursor-Position gültig wäre, spricht man von *Online-Validierung*.
Die bisher für Tunneldokumentationszwecke verwendeten Grafikeditoren (z. B. *AutoCAD* in *2doc*, siehe FURTMÜLLER&MARSCHALLINGER (2003)) scheitern daran, dass sie nicht explizit das Editieren von PSLGs und eine entsprechende Validierung unterstützen. Insbesondere erweist sich die Realisierung von Inseln beliebiger Schachtelungstiefe als problematisch. Solche Grafikeditoren bieten Funktionalität für eine Vielfalt allgemeiner Grafikobjekte. Das Editieren von PSLGs ist zwar prinzipiell möglich, jedoch unintuitiv und mit viel höherem Benutzeraufwand (im Sinne einer deutlich höheren Anzahl an erforderlichen Mausklicks) verbunden. Hinzu kommt die Gefahr eines nicht konsistenten und unangepassten zugrundeliegenden Datenmodells.

3 Implementierte Funktionalität

Die folgenden Operationen ermöglichen ein rasches Editieren und Modifizieren von Gebieten des PSLG und sind in unserer Bibliothek implementiert. (Standardoperationen wie z. B. Snapping sind auch verfügbar.) Eine Operation auf einem Gebiet bedeutet dabei eine Modifikation desjenigen Polygons, das dem Gebiet entspricht.

A Gebiet Erstellen

Diese Operation dient zum Zeichnen eines einfachen Polygons, welches den Rand einer neuen Zusammenhangskomponente (ZHK) des PSLG bildet. Dies kann entweder der Rand eines neuen PSLG oder eine neue Insel sein (siehe unten). Die Überschneidungsfreiheit (Einfachheit) des Polygons wird dabei explizit überprüft.

B Gebiet Erweitern

Entspricht dem Zeichnen einer überschneidungsfreien Polygonkette, deren Start- und Endpunkt am Rand einer bereits existierenden ZHK liegen. Das dem Rand entsprechende Polygon wird um die neu gezeichnete Polygonkette automatisch erweitert, wobei die vorher bestehende „Zwischenkette“ (in Abb. 4 und 5 strichliert eingezeichnet) gelöscht wird.

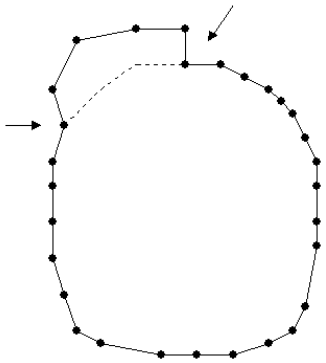


Abb. 4: Gebiet Erweitern (1)

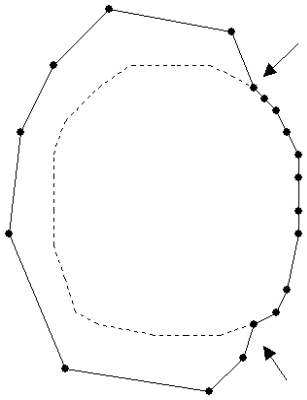


Abb. 5: Gebiet Erweitern (2)

C Gebiet Anhängen

Entspricht wieder dem Zeichnen einer Polygonkette mit Start- und Endpunkt am Rand einer ZHK. Jede innere Ecke der Kette liegt dabei außerhalb der ZHK. Diejenige Kette am Rand, welche die neugezeichnete Kette zu einem neuen Polygon ergänzt, wird automatisch gefunden - deren Ecken müssen also nicht explizit selektiert werden - und das so neu entstandene Gebiet wird in den PSLG integriert. Problematisch ist, wenn die Kanten der ergänzenden Kette Kanten jeweils verschieder Polygone sind, wie die strichlierte Kette in Abb. 6. Wenn bloß die Polygone welche die Fläche der ZHK zerteilen bekannt sind, dann ist es ohne das Wissen über ein strukturerhaltendes Polygon (i. e. dasjenige Polygon welches den Rand der ZHK, an die ein neues Polygon angehängt wird, darstellt) und den Beziehungen dieses Polygons zu den es zerteilenden Polygonen äußerst zeitaufwendig, herauszufinden, wie diese ergänzende Kette verläuft, und zu welchen Polygonen ihre Kanten gehören. (Die Theorie verwendet für den bei naivem Ausprobieren aller Möglichkeiten rasch anwachsenden Aufwand den Term *kombinatorische Explosion*.)

D Gebiet Teilen

Entspricht dem Zeichnen einer Polygonkette mit Start- und Endpunkt auf dem Rand eines beliebigen Gebietes, dem automatischen Erstellen eines neuen Gebietes (oder optional zweier neuer Gebiete) und der Integration dieses Gebietes in den PSLG (Abb. 7). Es reicht wiederum, nur diese teilende Polygonkette zu zeichnen; die sich so ergebenden Gebiete werden automatisch generiert. Weiters werden eventuelle Inseln automatisch diesen Gebieten korrekt neu zugeordnet.

E Insel Erstellen

Das Erstellen von Inseln entspricht dem Erstellen einer neuen ZHK im Inneren eines bereits bestehenden Gebietes. Eine Insel hat die gleichen Eigenschaften wie andere Gebiete des PSLG, d. h. es sind auch dieselben Operationen darauf anwendbar (z. B. *Anhängen*). Die Insel-Tiefe ist beliebig. Siehe Abb. 8 für ein Beispiel. eines PSLG mit Inseln beliebiger Tiefe.

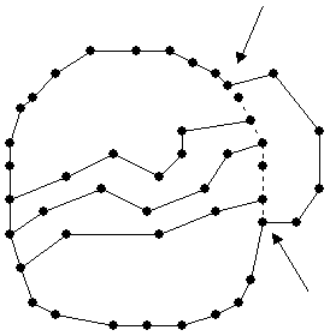


Abb. 6: Gebiet Anhängen

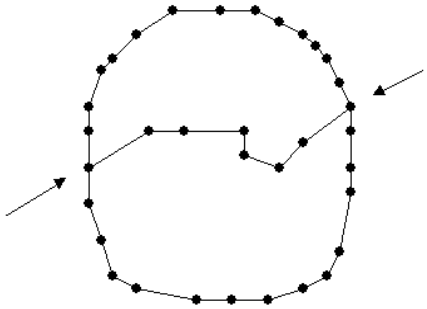


Abb. 7: Gebiet Teilen

F Ecken Verschieben

Der Benutzer selektiert eine Ecke und zieht sie via *Drag&Drop* auf die gewünschte neue Position. Dabei wird getestet, ob Schnittpunkte zwischen *inzidenten* Kanten[2] und den restlichen Kanten existieren. (Falls ja, dann ist die Operation ungültig.) Alle beteiligten Kanten und die betreffenden Polygone werden geeignet modifiziert (Abb. 9 und 10).
Neben der für die Laufzeit des Validierungsalgorithmus wichtigen Frage, welche Kanten notwendigerweise auf Schnitte überprüft werden müssen, ist ein weiteres Problem zu berücksichtigen: Obwohl keine Schnittpunkte entstanden sind, kann die Operation ohne weiterer Überprüfung zu einer geänderten Topologie des PSLG führen, die nicht automatisch erkannt wird. So könnte es z. B. dazu kommen, dass das Verschieben einer Ecke eines eine Insel enthaltenden Dreiecks dazu führt, dass die Insel nicht mehr im Inneren des Dreiecks liegt (Abb. 11 und 12).

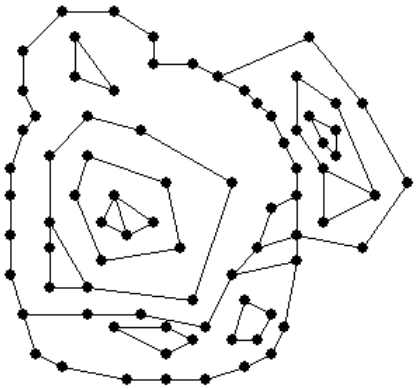


Abb 8: PSLG mit Inseln beliebiger Tiefe, die durch weitere Operationen (*Anhängen*, *Teilen*,...) modifiziert wurden.

Polygontypen

Das intern verwendete Datenmodell zur Repräsentierung eines PSLG ist eine hierarchische Datenstruktur von Polygonen. Sie besitzen spezielle Eigenschaften, gemäß derer sie in Typen klassifiziert werden. Dies erleichtert die Verwaltung des PSLG bei Validierung und Ausführung der Operationen. Im Folgenden werden die Polygontypen definiert:

***Kind*-Polygon:** Ein Kind-Polygon ist ein Polygon dessen Fläche sich vollständig innerhalb der Fläche eines andern, des sog. ***Eltern*-Polygons**, befindet. Dabei dürfen sich Eltern- und

[2] In einer Ecke sind Kanten des PSLG *inzident,* welche diese Ecke als Start- oder Endpunkt haben.

Kind-Polygon auch Kanten teilen[3]. Gemäß dieser Definition ist ein Kind immer Kind genau eines Eltern-Polygons, ein Eltern-Polygon kann jedoch mehrere Kinder haben.

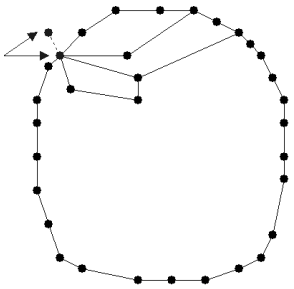


Abb. 9: Ecke Verschieben (1)

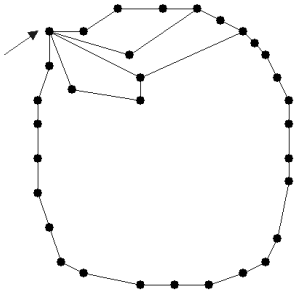


Abb. 10: Ecke Verschieben (2)

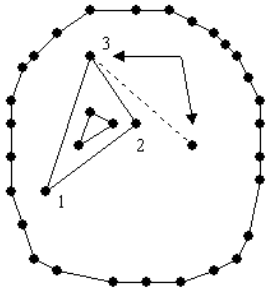


Abb. 11: Topologieänderung bei Operation Ecke Verschieben (1)

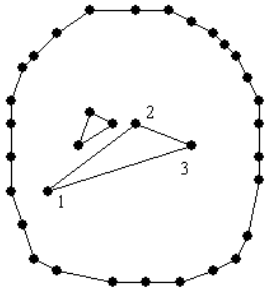


Abb. 12: Topologieänderung bei Op-eration Ecke Verschieben (2)

***Gebiets*-Polygon:** Es entspricht einem Gebiet des PSLG. Es hat eine eigene Fläche und modelliert die Geometrie einer Gesteinsfläche der Ortsbrust.

***Struktur*-Polygon:** Es ist das Gegenteil zu einem Gebietspolygon, da es keinem Gebiet des PSLG und damit keiner Gesteinsfläche entspricht. (Geologisch gesehen sind diese Polygone also redundant.) Ein Struktur-Polygon repräsentiert den Rand einer ZHK, deren Innenfläche durch mehrere Gebietspolygone zerteilt ist. Die Einführung von strukturerhaltenden Polygonen ist nötig zur Vereinfachung von Algorithmen zur Realisierung von Operationen wie *Gebiet Anhängen* (Abschnitt 3), um eine kombinatorische Explosion zu vermeiden.

***Komponentenrand*:** Dies ist ein Polygon welches die äußere Umgrenzung einer ZHK repräsentiert. Dessen Innenfläche kann durch mehrere Kinder-Polygone zerteilt sein. Der *globale Komponentenrand* ist die Außengrenze des PSLG, *lokale Komponentenränder* sind Außengrenzen von Inseln. Ein Komponentenrand kann (wie jedes andere Polygon) entweder Gebiets- oder Strukturpolygon sein.

[3] In diesem Fall ist das Eltern-Polygon ein *Struktur*-Polygon, das Kind-Polygon ein *Gedocktes* Polygon, siehe die nachfolgende Definition.

***Gedockte* Polygone**: Sie sind das Gegenteil von Komponentenrändern. Sie sind Gebietspolygone und Kinder eines strukturellen Komponentenrands.

Die Validierung von Operationen erfolgt neben dem Überprüfen auf Schnitten auf Basis der Typinformationen des Polygons. So ist z. B. die Operation *Gebiet Erweitern* bei Erweiterung eines Gebiets-Komponentenrands gültig, *Gebiet Anhängen* nur bei Anhängen des neuen Gebietes an einen Komponentenrand. Je nach Operation werden bei der Ausführung Polygontypen geändert, Polygone gelöscht, bzw. Polygone erstellt und in den PSLG integriert.

4 Geometrische Algorithmen für einen PSLG-Editor

Jede in Abschnitt 3 erläuterte Operation erfordert die Ausführung zumindest eines der folgenden geometrischen Algorithmen: Punkt-Lage-Bestimmung, Punkt-in-Polygon-Test, Entscheidung ob Schnitte in einer Menge von Geradensegmenten auftreten, bzw. ob Schnitte zwischen zwei Mengen von Geradensegmenten auftreten.
Angenommen, wir wollen die Ecke P des PSLG nach P' verschieben. Wenn die (in Abb. 13 schattierten) Dreiecke, die durch die in P inzidenten Kanten und P' aufgespannt werden, keine andere Ecke beinhalten und keine andere Kante des PSLG schneiden, dann ist auch die Verschiebung korrekt (Abb. 13).

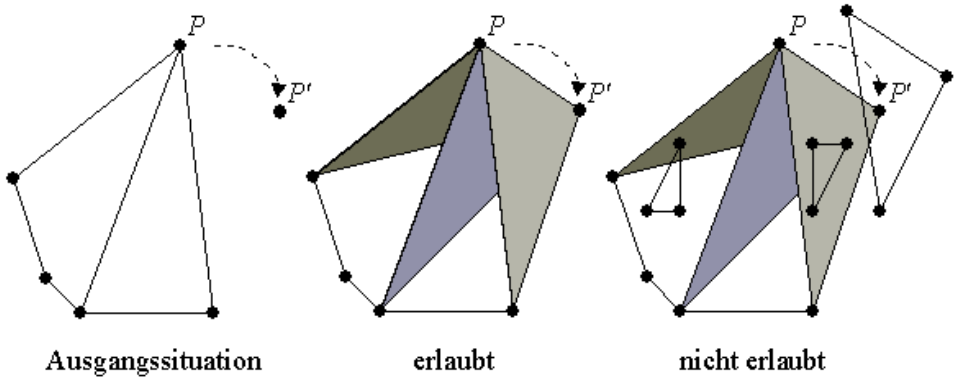


Abb. 13: Erlaubte und nicht erlaubte Verschiebung einer Ecke des PSLG

Bei Online-Validierung, also wenn während einer Cursor-Bewegung das Auftreten jeder unerlaubten Konstellation erkannt wird, ist diese Überprüfung während der Bewegung für die jeweils aktuelle Position erforderlich. (Online-Validierung, und damit der Zusatzaufwand vieler dynamischer Tests, ist daher deaktivierbar, insbesondere da sie auch hinderlich sein kann, wenn die Zielposition gültig ist, aber die Maus nur schwierig derart entlang eines Pfades vom Start zum Ziel bewegt werden kann, sodass dieser Pfad durchgehend nur erlaubte Konstellationen erzeugt.) Jedenfalls ist eine Menge von Geradensegmenten, nämlich die inzidenter Kanten, mit einer Menge anderer Geradensegmente (Kanten) auf Schnitt

zu testen, sowie das Enthaltensein von Ecken in einer Menge von Dreiecken zu testen. Für eine dynamische Überprüfung bei Online-Validierung sind diese Tests vielfach pro Verschiebung einer Ecke auszuführen.
Analog ist bei allen Editier-Operationen, die neue Geradensegmente generieren, eine derartige Schnittprüfung notwendig. Weiters ist z. B. beim Erstellen einer Insel notwendig, jenes Polygon des PSLG zu bestimmen, welches die neue Insel enthält, etwa durch Punkt-in-Polygon-Tests mit einer Ecke der Insel als Abfragepunkt.
Die Antwortzeit des PSLG-Editor hängt wesentlich davon ab, wie komplex die typischerweise zu verarbeitenden Eingabedaten sind und wie effizient diese geometrischen Probleme durch Algorithmen gelöst werden. Prinzipiell kennt die Algorithmische Geometrie für alle oben aufgeführten geometrischen Probleme Lösungsalgorithmen, welche optimal sind; siehe etwa (KLEIN 1997, O'ROURKE 1998). Dabei bedeutet *optimal*, dass die Anzahl der Schritte, die ein Algorithmus ausführt, größenordnungsmäßig gleich der gemäß Theorie mindestens erforderlichen Anzahl an Schritten ist. Das zentrale Ziel dieser Algorithmen ist, die Zahl der auszuführenden Schritte durch Anwendung von zusätzlichem mathematischen Wissen und passenden Datenstrukturen zu senken. So ist z. B. für den Test, ob *N* Geradensegmente sich schneiden, nicht notwendig, alle *N*(N-1)/2* vielen möglichen Paare von Geradensegmenten explizit auf Schnitt zu testen.
Die Situation bei unserem PSLG-Editor wird allerdings dadurch deutlich verkompliziert, dass die Daten, auf welche die geometrischen Algorithmen angewandt werden, nicht statisch sind: Zwischen den einzelnen Anwendungen der Algorithmen können sich die Position oder die Anzahl der Ecken und Geradensegmente des PSLG ändern. Die Theorie kennt zwar einerseits kinetische und andererseits dynamische Datenstrukturen, um eine der beiden Arten von Veränderungen von Daten zu modellieren, aber nur sehr wenig ist bekannt, um beide Arten von Veränderungen in beliebiger Reihenfolge zu modellieren. Zudem sind die dazu publizierten Ansätze sehr zeitaufwendig und diffizil zu implementieren.
Da bisher die Komplexität der zu verarbeitenden PSLGs auf wenige tausend Ecken beschränkt blieb, konnten wir mit einfacheren geometrischen Algorithmen das Auslangen finden, was bedeutet, dass ein Benutzer bei aktivierter Online-Validierung auf aktuellen PC-Systemen trotzdem keine wesentliche Verzögerung der Antwortzeit bemerkt.
Konkret nutzen wir die hierarchische Struktur der Daten, Typinformationen sowie das Wissen der Eingabepunkt-Lage aus, um Schnittprüfungen je nach Operation immer auf eine möglichst kleine Menge an Kanten einzuschränken. Etwa werden beim Verschieben einer Ecke einer Insel nur Kanten des Eltern-Polygons getestet, sowie Kanten, die zu Polygonen gehören, welche ebenfalls Kinder dieses Eltern-Polygons sind. Obwohl dies in der Theorie nicht verhindert, dass eventuell doch alle Kanten des PSLG getestet werden, führt dieses Vorgehen in der Praxis zu einer deutlichen Reduktion der auf Schnitt zu testenden Kantenpaare.
Analog nutzen wir bei Ermittlung der Eingabepunkt-Lage (z. B. beim Erstellen einer Insel) die hierarchische Struktur und führen solange rekursiv Punkt-in-Polygon-Tests nur für die Kinder des aktuell getesteten Polygons durch, bis das tatsächliche Eltern-Polygon gefunden wird, wodurch im günstigen Fall durchschnittlich in jedem Schritt die Hälfte der resultierenden zu testenden Polygone verworfen werden kann. Ein einzelner Punkt-in-Polygon-Test wird dabei gemäß des Algorithmus von Haines (O'ROURKE 1998) durchgeführt.

5 XML-Unterstützung

Zur persistenten Speicherung editierter PSLGs und damit der geometrischen Informationen einer Ortsbrust wird das Datenformat GML (Geographical Markup Language), eine Untersprache von XML verwendet. Jede dieser GML-Dateien steht dann z. B. für eine weitere Verarbeitung (z. B. Ableitung des 3D-Models) in einer Datenbank zur Verfügung. Die Vorteile sind: a) es ist ein standardisiertes und von anderen Grafikeditoren lesbares Format für Geo-Daten, b) es ist vom Menschen lesbar und c) XML wird von modernen, wirtschaftlich stark eingesetzten Programmiersprachen wie Java oder C# durch umfangreiche Bibliotheken unterstützt. Konkret wird ein PSLG abgespeichert als eine Liste von Polygonen, welche untereinander in Beziehung stehen. Strukturpolygone sind entsprechend markiert und notwendig, um den PSLG in der nächsten Editier-Session wieder korrekt zu laden und zur weiteren Bearbeitung zur Verfügung zu stellen. Für ein Beispiel siehe die erwähnte Webseite.

6 Diskussion und Ausblick

Diese Arbeit diskutiert die bei Editieroperationen in PSLG-Grafikeditoren mit Online-Validierung wichtigsten Probleme. Die derzeitige Implementierung ist zur Modellierung geologisch-geographischer Strukturen einsetzbar, denen PSLGs mit maximal einigen hundert Ecken entsprechen, wie z. B. Tunnelquerschnitte. Hinsichtlich der Leistungsfähigkeit der Algorithmen sind im Moment die Algorithmen zur Punkt-Lage-Bestimmung in einem PSLG und zum Finden von Schnitten in Mengen von Geradensegmenten am kritischsten. Aufgrund der modularen Implementierung können die zugrundliegenden geometrischen Algorithmen bei Bedarf an die Komplexität der typischerweise auftretenden Datenmengen angepasst werden. Eine entsprechende Verbesserung ist Gegenstand künftiger Weiterentwicklungen. Da ein auf der Bibliothek basierendes Tunneldokumentationssystem nicht an *AutoCAD* gebunden ist und Eingabefehler aufgrund von Online-Validierung frühzeitig erkannt werden, erwarten wir eine höhere Akzeptanz bei Geologen. (Die 3D-Konstruktion bzw. die Generierung der 3D-Daten wird weiterhin in *AutoCAD* durchgeführt werden.) Natürlich ist die Algorithmenbibliothek nicht nur auf den Einsatz in einem Tunneldokumentationssystem beschränkt, sondern es sind weitere GIS-Applikationen vorstellbar, wie z. B. Editoren für Versorgungsnetzwerke.

7 Literatur

Klein, R. (1997): *Algorithmische Geometrie*. Addison-Wesley-Longman-Verlag.

O'Rourke, J. (1998): *Computational Geometry in C.* Cambridge University Press, second edition. ISBN 0521649765.

ÖGG (Hrsg., Oktober 2001): *Richtlinie für die Geomechanische Planung von Untertagebauarbeiten mit zyklischem Vortrieb. Gebirgscharakterisierung und Vorgangsweise zur nachvollziehbaren Festlegung von bautechnischen Maßnahmen während der Planung und Bauausführung*. Österreichische Gesellschaft für Geomechanik, Salzburg.

Furtmüller, G. & Marschallinger, R. (2003): *Tunneldokumentation mittels 2doc – zeitgemäßes Management komplexer Geodaten beim Tunnelvortrieb*. Felsbau 21(3):17–20.