

## Prozeßsimulation und -optimierung am Beispiel der Fertigung von Zahnkantenabdachungen

Heimannsfeld, K.; Schiedeck, N.

*Aufgrund der komplexen Prozeßkinematik ist die Konstruktion von fertigen Zahnkantenabdachungen schwierig. In einem ersten Ansatz wurde die Berechnung durch eine CAD-basierte Simulation realisiert. Damit konnte die notwendige Zeit zur Berechnung der Prozeßparameter von mehreren Tagen auf 4-8 Stunden verkürzt werden. Dafür waren bisher ein teures, parametrisches 3D CAD-System und ein erfahrener Konstrukteur nötig. Die Erfahrungen aus der CAD-basierten Simulation wurden zur Entwicklung eines automatisierten Berechnungs- und Optimierungsprogramms genutzt, so daß die Auslegungszeit auf 1 Stunde reduziert werden konnte. Die Arbeiten erfolgten in Kooperation mit der Robert Bosch GmbH in Hildesheim.*

*Due to the kinematics of the manufacturing process the design of toothed chamfers is a very complex task. In a first approach a CAD-based simulation was used. The design time for the tool parameters were reduced from a few days to 4-8 hours. Unfortunately this requires an expensive parametric 3D CAD-system and an experienced designer. The experience gained from the design of the CAD-simulation was applied to the design of a new fully automatic optimizer. This approach reduced the design time to approx. 1 hour. This research was conducted in cooperation with Robert Bosch GmbH, Hildesheim.*

### 1 Einleitung und Problembeschreibung

An Zahnrädern, die Schaltvorgänge ausführen müssen, werden die Zahnkanten abgedacht. Dabei haben sich je nach Einsatz der Zahnräder unterschiedliche Abdachformen als geeignet erwiesen. Für die Fertigung werden neben umformenden Verfahren auch spanende Verfahren eingesetzt (s. **Bild 1**).

An Starterritzeln befinden sich zwei asymmetrisch angeordnete Abdachungen, die mit einem zweischneidigen Fräs Werkzeug hergestellt werden. Das Verfahren besitzt eine Anzahl von variablen Parametern, z. B.

die Einstellung der Fräserachse im Raum, die Startwinkel von Werkzeug- und Werkstückspindel etc. Weitere variable Parameter besitzt das Werkzeug, an dem die Position der Hauptschneiden variiert werden kann.

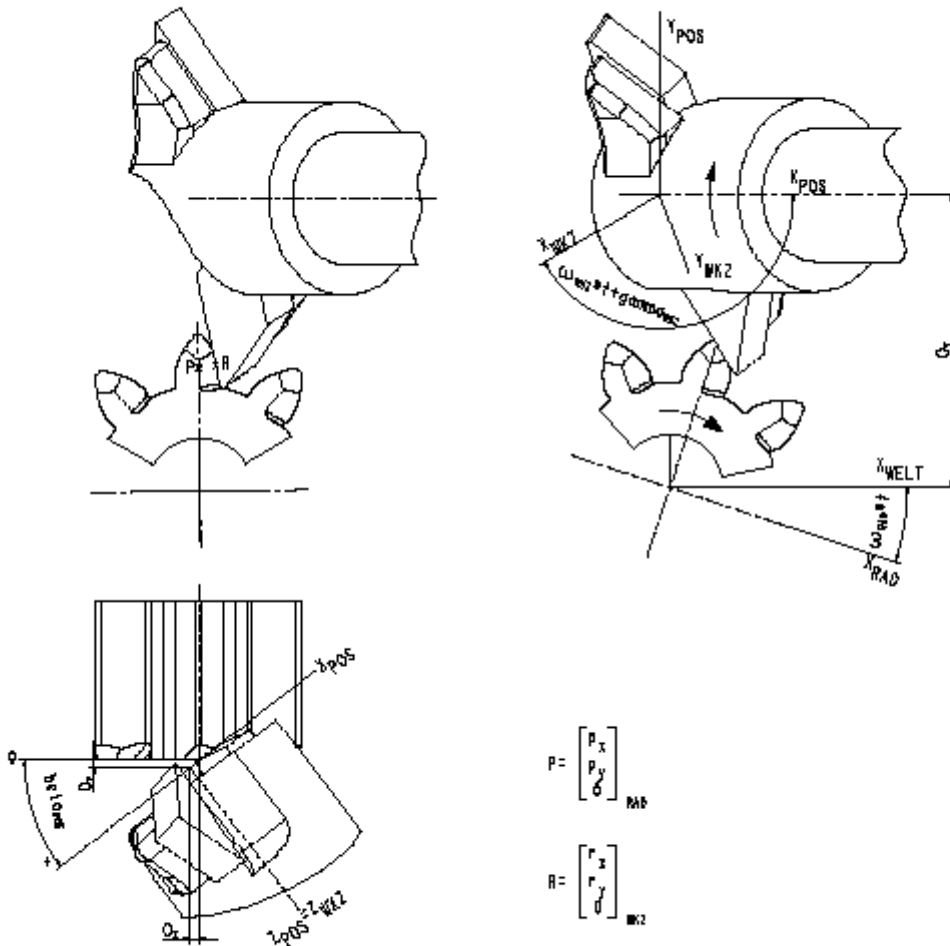
Das anwendungstechnische Hauptproblem besteht in der Ermittlung von geeigneten Werten für die genannten Prozeßparameter. Bisher wurde diese Aufgabe mit Hilfe einer CAD-basierten Simulation /1/ gelöst. Damit kann das Fertigungsergebnis bei unterschiedlichen Prozeßeinstellungen simuliert werden. Je nach Geschick und Erfahrung des Anwenders konnte innerhalb eines Zeitrahmens von 4 bis 8 Stunden eine geeignete Einstellung gefunden werden.

Die Robert Bosch GmbH stellt eine sehr hohe Anzahl von unterschiedlichen Starterritzeln her. Viele sind für Nutzfahrzeuge und werden nur in kleiner oder mittlerer Stückzahl hergestellt. Für die Fertigung dieser Teilefamilie sollte ein Softwarewerkzeug entwickelt werden, mit dem die geeigneten Maschinen- und Werkzeugeinstellungen berechnet werden können. Die Berechnung soll vollkommen automatisch ablaufen, eine manuelle Iteration war unter anderem auch aus Zeitgründen nicht akzeptabel. Da im Fertigungsbereich keine CAD-Systeme vorhanden sind, mußte ein eigenständiges Berechnungsprogramm implementiert werden.

### 2 Prozeßsimulation und -optimierung

Als Lösungskonzept wurde eine Optimierung mit Hilfe eines Simulationsmodells gewählt. Eine analytische Lösungsfindung ist aufgrund der Komplexität der geometrisch-kinematischen Zusammenhänge nicht durchführbar (s. **Bild 2**).

Das Konzept besteht daraus, daß mit einem Startvektor für die Prozeß- bzw. Optimierparameter eine Prozeßsimulation durchgeführt wird. Durch Auswertung bestimmter Optimierkriterien erzeugt ein Optimierungsalgorithmus eine neue, besser geeignete Generation



**Bild 1:** Kinematische Optimierparameter des Zahnkantenabdachprozeß

von Optimierparametern, die zur Simulation genutzt werden. Sind die Sollwerte für die Optimierkriterien innerhalb der definierten Toleranzen, so bricht diese Optimierschleife ab und die jüngste Generation von Prozeßparametern wird als Ergebnis ausgegeben (s.**Bild 3**).

Der Realisierung des Systems ist eine ausführliche

Modellierung und Spezifikation vorangegangen. Zu Projektbeginn wurden für alle relevanten kinematischen Prozeßmerkmale eindeutige Bezeichnungen festgelegt. Großen Aufwand erforderte das Aufstellen und Validieren der Gleichungen für die Gestaltsberechnungen.

Anschließend erfolgte die Festlegung der Optimierkriterien, mit denen das Simulationsergebnis bewertet wird. Sie wurden identisch mit den Gestaltsparametern für die Zahnkantenabdachungen gewählt, die in den Konstruktionszeichnungen vorgegeben sind. Gleichzeitig wurden die (variablen) Optimierparameter ausgewählt. Nach diesen Arbeiten mit mehr vorbereitenden Charakter wurde durch systematische Untersuchungen am Simulationsmodell herausgefunden, welchen Einfluß die (prozeßbezogenen) Optimierparameter auf die (gestaltsbezogenen) Optimierkriterien besitzen. Aufbauend auf

$$\begin{aligned}
 R^h x_{RAD} &= -\text{Sin}(t \omega_{RAD}) \left( O_y - l_{Rh} \text{Cos}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) + \right. \\
 &\quad \left. \text{Versatz}_{Rh} \text{zeich Sin}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) \right) \\
 &\quad + \text{Cos}(t \omega_{RAD}) \left( O_x + \text{Cos}\beta_{POS} \right. \\
 &\quad \left. \text{Versatz}_{Rh} \text{zeich Cos}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) + \right. \\
 &\quad \left. l_{Rh} \text{Sin}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) \right) \\
 R^h y_{RAD} &= \text{Cos}(t \omega_{RAD}) \left( O_y - l_{Rh} \text{Cos}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) + \right. \\
 &\quad \left. \text{Versatz}_{Rh} \text{zeich Sin}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) \right) \\
 &\quad + \text{Sin}(t \omega_{RAD}) \left( O_x + \text{Cos}\beta_{POS} \right. \\
 &\quad \left. \text{Versatz}_{Rh} \text{zeich Cos}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) + \right. \\
 &\quad \left. l_{Rh} \text{Sin}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) \right) \\
 R^h z_{RAD} &= O_z + \text{Sin}\beta_{POS} \left( \text{Versatz}_{Rh} \text{zeich Cos}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) + \right. \\
 &\quad \left. l_{Rh} \text{Sin}(\gamma_{WKZ} + t \omega_{WKZ} \text{zeich}) \right)
 \end{aligned}$$

**Bild 2:** Koordinatengleichungen der Hüllfläche, die die Rhombusschneide relativ zum Werkzeug beschreibt.

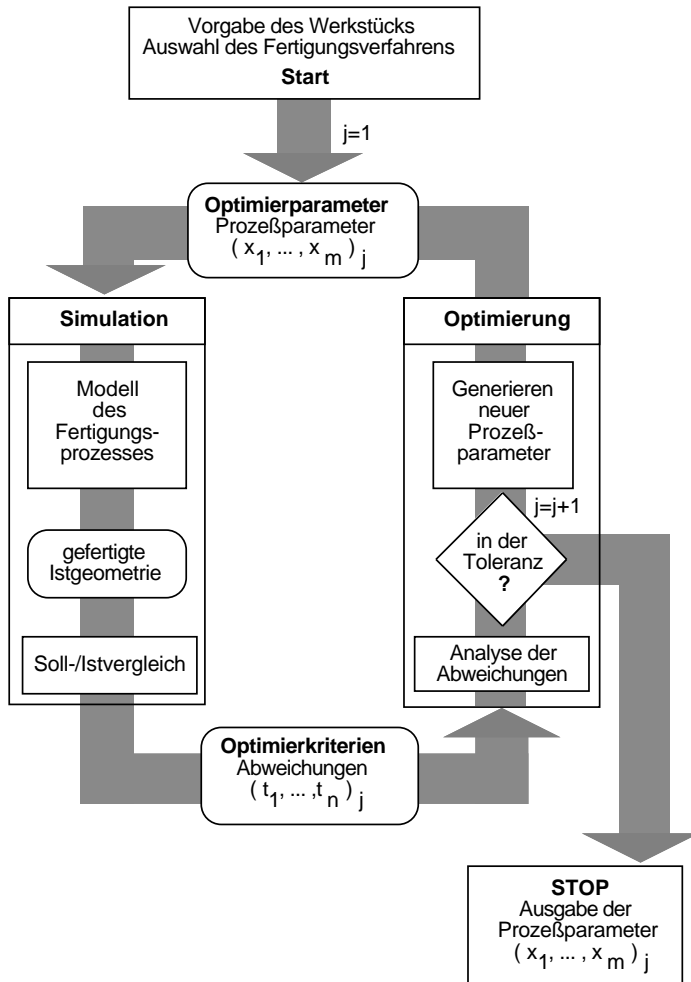


Bild 3: Konzeptioneller Aufbau eines Simulation- und Optimierungssystem

PC lauffähiges Programm. Da die Größe des Projektes aber mit ca. 10000 Zeilen (3/4 Mannjahre) für die Implementierung des Optimierers angesetzt wurde, stellte die Vorgabe von Fortran 77 eine erhebliche Einschränkung der Modularisierbarkeit der Software dar. Eine Softwareentwicklung nach dem heutigen Stand der Technik wäre nicht möglich gewesen. Auch die Implementierung einer komfortablen Benutzerschnittstelle für den Anwender wäre unter Fortran 77 nur bedingt möglich gewesen.

Da für eine teamorientierte Entwicklung von Software unter dem Zielbetriebssystem Windows NT keine geeignete Software zur Verfügung stand, wurde die komplette Programmentwicklung auf den im Institut vorhandenen SUN Solaris Workstations durchgeführt. Die damit notwendige Portabilität verbot natürlich die Nutzung von systemspezifischen Bibliotheken (z.B. QuickWin von Microsoft) zur Implementierung der Benutzerschnittstelle. Deswegen wurde in Absprache mit der Robert Bosch GmbH das in Bild 4 dargestellte Systemkonzept gewählt.

Das Gesamtsystem wird in zwei unabhängige Teilsysteme gegliedert, die mit unterschiedlichen Werkzeugen realisiert werden und über die von den verschiedenen betriebssystemspezifische Mechanismen kommunizieren. Glücklicher Weise unterstützt Windows NT ei-

diesen Erkenntnissen konnten Optimierbausteine für die Optimierung jeweils eines Kriteriums entwickelt werden, aus denen sich der Optimieralgorithmus aufbaut. Die vollständige Entwicklung des Optimieralgorithmus erfolgte am bereits realisierten Softwaresystem. Insbesondere die Spezifikation von korrekten Abbruchkriterien interner Iterationsschleifen erforderte eine beachtliche Versuchsphase.

### 3 Implementierung

Die Hauptanforderung der Robert Bosch GmbH bestand in einer Implementierung in Fortran 77 und ein auf einem Industrie

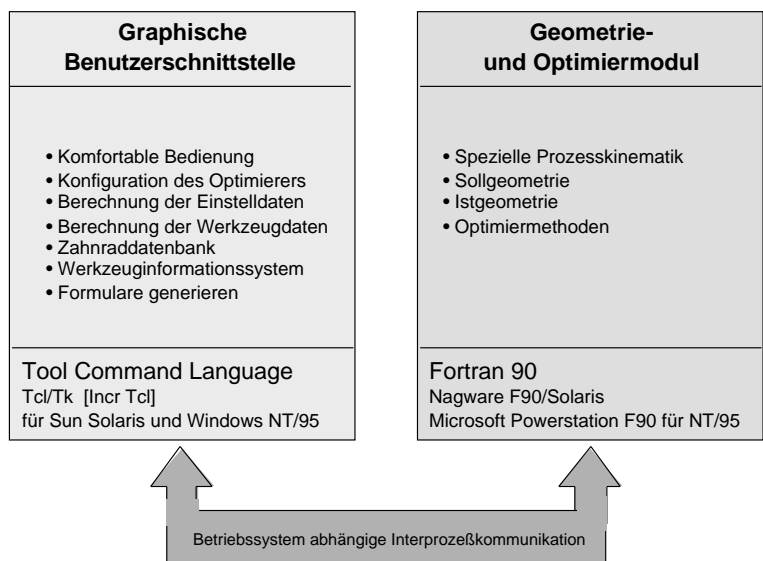


Bild 4: Systemkonzept der Implementierung

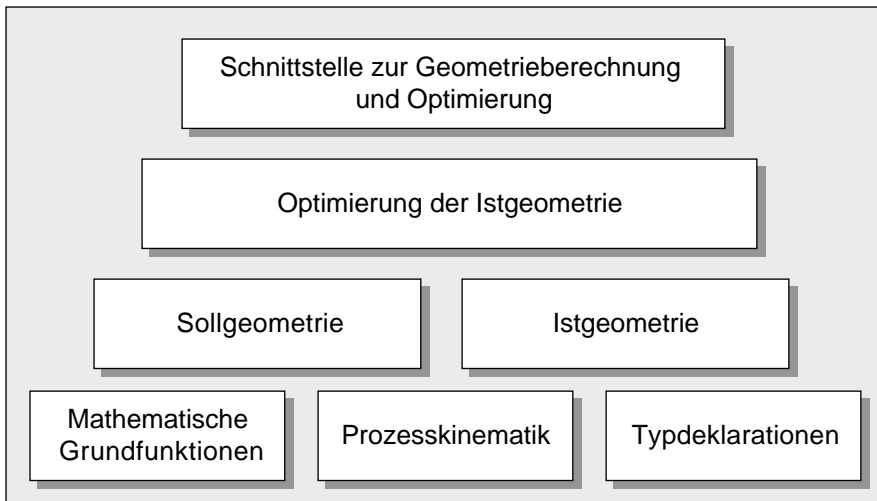


Bild 5: Modulstruktur Optimierer

ne dem UNIX sehr verwandte Prozeßkommunikation, so daß der Portierungsaufwand gering ist.

Durch die Trennung der Benutzerschnittstelle von dem eigentlichen Berechnungs- und Optimieralgorithmus kann jederzeit einer der Teile weiterentwickelt werden, ohne das weitere Veränderungen an dem anderen Programmteil notwendig sind. Dadurch kann zu einem späteren Zeitpunkt z.B. eine Kopplung an ein PPS-System erfolgen, ohne daß der Optimierer geändert werden muss.

### 3.2 Realisierung des Optimierers

Zur Implementierung des Berechnungs- und Optimiermodul wurde die Programmiersprache Fortran 90 /2,3,5/ gewählt. Fortran 90 erweitert das im wissenschaftlichen Umfeld häufig eingesetzte Fortran 77 um Konzepte zur Strukturierung und Modularisierung. Viele der neue Sprachkonstrukte sind aus den Programmiersprachen C und Modula-2 übernommen worden. Von der Realisierung in der Programmiersprache C wurde wegen der oft verwirrende Zeigerkonstrukte zur Implementierung von Matrixoperationen Abstand genommen.

**Bild 5** zeigt die Modulstruktur der Implementierung. Insgesamt entstanden 10 verschiedene Module, die über ihre Schnittstellen eine saubere Trennung der verschiedenen Berechnungsebenen und -funktionen realisieren. Insgesamt umfassen alle Module ungefähr 8000 Programmzeile mit 264 Unterprozeduren und Funktionen.

Die unterste Ebene der Modulstruktur bilden die ma-

thematischen Grundfunktionen, die nicht in den bereits umfangreichen Bibliotheken der Fortran 90 Entwicklungs-umgebung enthalten sind, spezielle Funktionen zur Berechnung der Kinematik bei der Herstellung der Zahnkantenabdachungen und die Datentypdeklarationen. Dabei enthalten die Module zur Datentypdeklaration nicht nur die eigentliche Deklaration von einfachen und zusammengesetzten Datentypen,

sondern auch die Zugriffsfunktionen. Damit wird eine Kapselung aller Datenstrukturen von der eigentlichen Implementierung erreicht.

Ein wesentlicher Anteil der Arbeiten an den mathematischen Grundfunktionen hatte die Entwicklung von stabilen Verfahren zur numerischen Lösung von Gleichungen und von allgemeinen, stabilen Methoden zur ein- und mehrdimensionalen Optimierung. Viele der in der Literatur dokumentierten Verfahren erwiesen sich in diesem Anwendungsfall als instabil und problematisch.

Die zweite Modulschicht ist für die Berechnung der Soll- und Istgeometrien der gefertigten Zahnradabdachungen zuständig. Sie greift über die definierten Schnittstellen auf die Gleichungslösungsverfahren, die definierten Datenstrukturen und die speziellen kinematischen Funktionen zu. Durch die Trennung von Schnittstelle und Implementierung konnten nachträglich notwendig gewordene Änderungen in der Berechnung der Zahnradgeometrie ohne Probleme und ohne Nebeneffekte auf andere Programmteile durchgeführt werden.

Die Funktionen zur Optimierung bilden die dritte Schicht der Implementierung. Unter Benutzung der Differenz der Sollgeometrie und der Istgeometrie berechnet eine anwendungsspezifische Zielfunktion die geometriebestimmenden Parameter. Der Versuch einen allgemeinen, mehrdimensionalen Optimierungsalgorithmus zu verwenden ergab nur unbefriedigende Ergebnisse. Dies ist auf die im Zielbereich sehr flache und mit vielen lokalen Extrema versehenen Optimierungsfunktion zurückzuführen.

Die Portierung der Fortran 90 Quellprogramme von dem Entwicklungssystem unter *SUN Solaris Workstations* auf das Zielsystem *Windows NT* mit dem Microsoft Compiler *Fortran PowerStation* erwies sich trotz aller Vorbehalte als unproblematisch. Die einzigen Probleme ergaben sich bei der unterschiedlichen Benennung von systemspezifischen Modulen, die zur Realisierung der Interprozeßkommunikation verwendet wurden.

### 3.3 Realisierung der Benutzerschnittstelle

Die Realisierung der Benutzerschnittstelle war ursprünglich in Fortran 90 geplant. Da in Fortran 90 die Entwicklung einer einfachen textuellen Oberfläche denselben Aufwand wie die Implementierung einer graphischen Benutzeroberfläche mit spezialisierten Werkzeugen bedeutet, wurde die Programmiersprache Tcl/Tk (Tool Command Language/Toolkit) /4/ ausgewählt. Tcl ist eine interpretierte Programmiersprache, die insbesondere zur schnellen Entwicklung von graphischen Benutzeroberflächen entworfen wurde. Tk ist eine Bibliothek die wichtige Grund-

lemente zusammenfaßt und als Grundlage für komplexere anwendungsspezifische Benutzeroberflächen zur Verfügung stellt.

Tcl/Tk stammt aus dem universitären Umfeld wird heute aber von der Firma Sun Microsystems weiterentwickelt und frei zur Verfügung gestellt. Im Rahmen ihrer Internetaktivitäten wird Tcl/Tk ergänzend zu Java als Internet Scripting Language propagandiert. Mittlerweile existieren sogar sogenannte *Plugins* für Internetbrowser mit denen Tcl/Tk Applikationen innerhalb eines HTML Dokument ausgeführt werden können. Durch die daraus resultierende notwendige Verfügbarkeit ist Tcl/Tk für alle gängigen Computerplattformen erhältlich. Im Rahmen dieses Projektes wurde ferner auch die objektorientierte Erweiterung [incr Tcl] von Tcl/Tk genutzt, um auf die erweiterten Klassenbibliotheken von Fensterelementen zurückzugreifen.

Die Benutzerschnittstelle umfasst ca. 2500 Programmzeile und wurde mit einem Aufwand von ca 1 Mannmonat implementiert. Der erhebliche Zeitun-

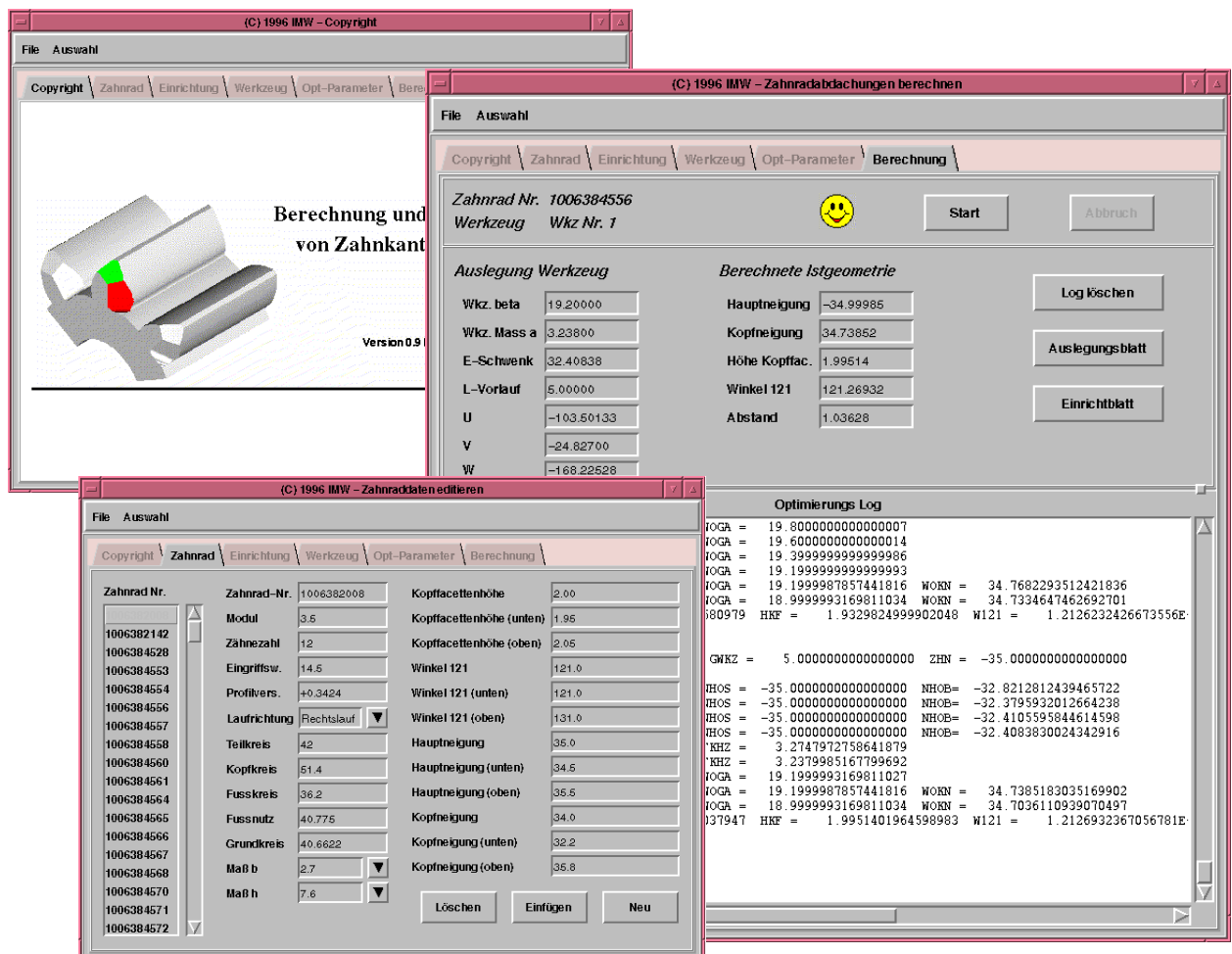


Bild 6: Die graphische Benutzeroberfläche

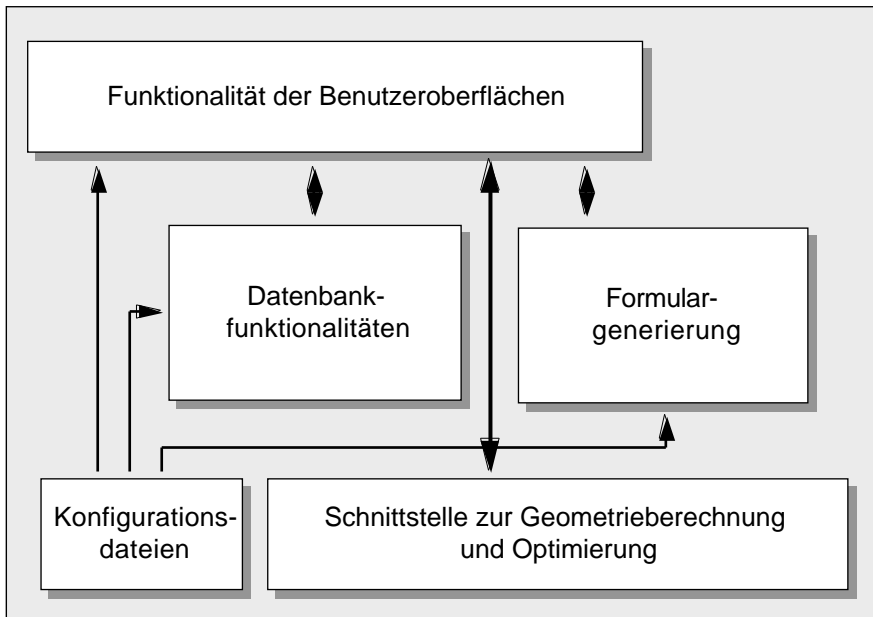


Bild 7: Modulstruktur Benutzeroberfläche

terschied in dem Verhältnis zwischen Zeile/Zeit zwischen den Fortran 90 Modulen und der Benutzerschnittstelle resultieren hauptsächlich aus den oftmals sehr komplexen und dichten mathematischen Quellcode, der einen erheblichen Aufwand bei der Programmierung und der Validierung erfordert. **Bild 6** zeigt die verschiedenen Elemente der graphischen Benutzeroberfläche.

Die Benutzerschnittstelle wurde in vier Teilbereiche gegliedert. **Bild 7** zeigt die Struktur der graphischen Benutzerschnittstelle. Der größte Teil stellt hier bei die eigentliche Realisierung der Bedienungselemente und deren Funktionalität dar. Ungefähr 1000 Zeilen und damit 50% der Programmieraufwandes entfielen auf dieses Modul. Zur Verwaltung der Zahnrad- und der Werkzeugdaten wurde eine einfache Datenbank und die benötigten Zugriffsfunktionen implementiert. Das Datenbankmodul enthält auch alle Funktionen zum Speichern, Laden und Konvertieren von Dateien.

Auf besonderen Wunsch wurde als drittes Modul eine automatische Generierung von Auslegungsformularen für die Fräswerkzeuge und die Einrichtdaten der Werkzeugmaschinen realisiert. Ausgehend von einer Formularkonfigurationsdatei und einer Formulervorlage in der Seitenbeschreibungssprache *Postscript* können die zwei Formulare nach erfolgreichem Abschluß der Optimierung automatisch generiert, angezeigt und ausgedruckt werden. Die In-

halte des Formulars können dabei über die Konfigurationsdatei leicht verändert und angepasst werden.

Das vierte Modul stellt die Verbindung zwischen dem Optimierer und der Benutzeroberfläche her. Dabei wurde eine echte programmtechnische Parallelität realisiert, so daß mit der Datenbank gearbeitet werden kann, während eine Optimierung im Gange ist.

#### 4 Zusammenfassung

Durch die Integration von Simulation und Optimierung konnte erstmals ein vollständig automatisiertes Verfahren zur Auslegung und Berechnung der Prozeßparameter bei der Fertigung von Zahnkantenabdachungen realisiert werden. Dabei hat sich gezeigt, daß ein wesentlicher Punkt bei dem Einsatz von Optimierungsverfahren die Einschränkung des mehrdimensionalen Suchraums d.h. die Verwendung von anwendungsspezifischen Wissen ist. Dies ist insbesondere für die Auswahl eines geeigneten Startpunktes der Optimierung notwendig.

#### 5 Literatur

- /1/ Schiedeck, N.: Zahnkantenabdachungen - Fertigungsgerechte Gestaltung und Prozeßauslegung mit CAD-basierter Simulation. Mitteilungen Nr.20 Institut für Maschinenwesen, 1995
- /2/ Brainerd, W. S. et al.: Fortran 90, Oldenburg (1991)
- /3/ Metcalf, M. & Reid J.: Fortran 90 explained, Oxford University Press (1991)
- /4/ Ousterhout J.K.: Tcl und Tk, Entwicklung grafischer Benutzerschnittstellen, Addison-Wesley (1995)
- /5/ N.N.: Microsoft Fortran 90 Powerstation, Reference Guide, Language Guide (1995)