

# Java Optik

## Interaktive Java-Applets zur geometrischen Optik

Marcel Schmittfull

Juni 2002

Diese Arbeit beschäftigt sich mit der Entwicklung interaktiver Java Applets, die Gesetzmäßigkeiten und Phänomene der geometrischen Optik simulieren. Es sollen Schülern, Studenten und Interessierten die Grundprinzipien der geometrischen Optik möglichst interaktiv und anschaulich vermittelt und Fortgeschrittenen die Möglichkeit zur schnellen Berechnung optischer Zusammenhänge gegeben werden.

# Inhaltsverzeichnis

<b>1 Programmiersprachen</b>	<b>1</b>
1.1 Applets . . . . .	1
1.2 Geometria/Geoscript . . . . .	1
1.3 HTML, JavaScript und CSS . . . . .	1
1.4 Browserunterstützung . . . . .	2
1.5 Systemvoraussetzungen . . . . .	2
<b>2 Layout</b>	<b>2</b>
2.1 Allgemein . . . . .	2
2.2 Applets . . . . .	2
2.3 Weg der Daten . . . . .	3
<b>3 Benutzerführung</b>	<b>3</b>
3.1 Begleittexte . . . . .	3
3.2 Navigation und Orientierung . . . . .	3
3.3 Fehlermanagement . . . . .	4
<b>4 Didaktische Aspekte</b>	<b>4</b>
4.1 Ersatz für reale Experimente . . . . .	4
4.2 Begleittexte . . . . .	4
4.3 Überwachung . . . . .	5
<b>5 Ähnliche Produkte</b>	<b>5</b>
<b>6 Danksagung</b>	<b>6</b>
<b>A Aktueller Entwicklungsstatus</b>	<b>7</b>
<b>B Dateistruktur</b>	<b>7</b>
<b>C Geometria/Geoscript</b>	<b>7</b>
<b>D Jugend forscht – Schüler experimentieren</b>	<b>8</b>

# 1 Programmiersprachen

## 1.1 Applets

Java-Applets<sup>1</sup> haben gegenüber herkömmlichen Programmen einige Vorteile, weil

- sie in HTML-Dokumente eingebettet werden können,
- sie somit direkt in einem Browser betrachtet und bedient werden können,
- der jeweils verwendete Browser dem Benutzer vertraut ist,
- sie on- und offline abrufbar und ausführbar sind,
- keine Installation erforderlich ist,
- sie plattformunabhängig sind, also sowohl von Windows-, als auch von Linux- und Unix-Betriebssystemen unterstützt werden.

## 1.2 Geometria/Geoscript

Die Applets wurden mit der Programmiersprache Geometria/Geoscript<sup>2</sup> von Dr. Timo Ehmke<sup>3</sup> geschrieben. Diese Programmiersprache erlaubt es, sehr komplexe Geometrie-Konstruktionen als Applets darzustellen, wobei die gesamte Konstruktion in sich sehr flexibel ist.

So kann der Benutzer in den Applets bestimmte Punkte mit der Maus ziehen und die gesamte Konstruktion passt sich den Veränderungen umgehend an. Kombiniert mit einigen Schieberegler ergibt sich somit eine ungeheuer große Flexibilität der Applets<sup>4</sup>. In manche Applets wurden zudem Bilder, z.B. von Linsen, direkt eingefügt, wobei auch diese Bilder flexibel sind, d.h. sie passen sich der Konstruktion bei Veränderungen an und können sogar selbst mit der Maus verschoben werden. Die Applets können außerdem mit JavaScript kommunizieren<sup>5</sup>. So können v.a. komplizierte Passagen, die mit Geometria/Geoscript nicht realisierbar sind, einfach mit JavaScript gelöst werden, indem das jeweilige Applet einige Werte an eine JavaScript-Funktion leitet, diese JavaScript-Funktion die gewünschten Ergebnisse berechnet und danach die Endwerte wieder an das jeweilige Applet zurückgibt. Solche Funktionen wurden in beinahe jedem Applet verwendet; ein Beispiel befindet sich in der Datei `ap/tore/tore.js`.

Nähere Informationen zum Aufbau und zur Syntax von Geometria/Geoscript befinden sich in Anhang C.

## 1.3 HTML, JavaScript und CSS

Es wurden in der Programmiersprache HTML ausführliche, für einen Gymnasiasten der Mittelstufe nachvollziehbare Begleittexte zur geometrischen Optik und zu den Applets geschrieben. Die Navigation, die Orientierung und der Datenaustausch zwischen den HTML-Seiten er-

---

<sup>1</sup>„Java Applets“ sind Programme, die in der Programmiersprache Java geschrieben sind und in HTML-Dokumente eingebettet werden können (siehe <http://java.sun.com/>).

<sup>2</sup><http://www.geometria.de/>

<sup>3</sup>Institute for Science Education (IPN), University of Kiel

<sup>4</sup>In manchen Applets dieser Arbeit sind über sechs Punkte frei bewegbar und zusätzlich fünf Werte mit Schieberegler einstellbar !

<sup>5</sup>Genauer: Die Programmiersprache Geometria/Geoscript kann mit JavaScript kommunizieren.

folgen hauptsächlich durch JavaScript, CSS (Cascading-Style-Sheets) beeinflussen das Design und Layout der HTML-Seiten wesentlich.

## 1.4 Browserunterstützung

Bei der Programmierung in HTML, JavaScript und CSS wurde sehr darauf geachtet, dass sowohl der Internet Explorer ab Version 4.0, als auch der Netscape Navigator ab Version 4.0 alle Seiten möglichst gut und fehlerfrei darstellen kann.

## 1.5 Systemvoraussetzungen

Um die Applets betrachten zu können ist die Java-Virtual-Machine ab Version 1.1.5 (z.B. Netscape Communicator 4.0.6, Internet Explorer 4.0, HotJava 1.1.5) erforderlich. Die Hardware-Mindestvoraussetzungen sind 16 MB Arbeitsspeicher und 166 MHz Prozessor. Zudem dürfen Java-Applets im Browser nicht deaktiviert sein.

# 2 Layout

## 2.1 Allgemein

Alle zum Projekt gehörenden HTML-Seiten, also Begleittexte, Übersichten, Info-Seiten, etc., werden im Start-Frameset<sup>6</sup> angezeigt<sup>7</sup>. Dieses Frameset ist in drei Teile bzw. Frames gegliedert<sup>8</sup>: Das linke Frame enthält das Menü, das Frame rechts oben zeigt den Pfad der jeweils gerade aufgerufenen HTML-Seite und eine kurze Beschreibung des jeweils mit der Maus ausgewählten Links. In dem großen Frame darunter wird der eigentliche Inhalt ausgegeben.

Beim Design der HTML-Dokumente wurde sehr viel Wert auf gute Lesbarkeit und Übersichtlichkeit gelegt. Erreicht wurde dies v.a. durch CSS (siehe die Datei `styles.css`).

## 2.2 Applets

Die HTML-Seiten, die die Applets beinhalten, werden als einzige Ausnahme nicht ins Startframeset, sondern in ein neues Browserfenster, das ein in drei Zeilen unterteiltes Frameset enthält, geladen. Zur Optimierung des Datenaustausches zwischen den Frames (siehe Abschnitt 2.3) wurde in diesem Frameset zusätzlich noch ein verstecktes viertes Frame definiert.

Das obere Frame enthält einige Links zu Seiten des Projektes, die mit dem im Applet behandelten Thema verwandt sind<sup>9</sup>. Darunter wird eine kurze Beschreibung des jeweils ausgewählten Links angezeigt. Das mittlere Frame zeigt das jeweilige Applet selbst und einen kurzen Einführungstext zur Funktionsweise dieses Applets. Im unteren Frame befinden sich ausführliche Begleittexte<sup>10</sup>.

---

<sup>6</sup>Ein Frameset teilt das Browserfenster in mehrere „Unterfenster“ (Frames) ein. Der HTML-Code zu dieser Einteilung befindet sich in der Datei `frames.html`.

<sup>7</sup>Die einzige Ausnahme stellen die Applets selbst dar (siehe Abschnitt 2.2).

<sup>8</sup>Eigentlich besteht das Frameset aus fünf Frames. Weil das `menubottom`-Frame unter dem Menü-Frame jedoch für den Benutzer nicht als eigenes Frame erkennbar ist und weil das völlig unsichtbare `hidden`-Frame nur zu Programmierzwecken dient, wird nur auf drei Frames eingegangen.

<sup>9</sup>Beispielsweise beim Totalreflexion-Applet eine Erklärung des Reflexionsgesetzes und des Brechungsgesetzes, Downloadmöglichkeiten des Totalreflexion-Applets, ...

<sup>10</sup>Quelltext siehe z.B. in der Datei `ap/tore/tore_guide.js`.

Das Niveau dieser Begleittexte kann der Benutzer selbst wählen. Die Texte werden zudem nicht auf einmal, sondern stückweise ausgegeben, sodass der Benutzer durch den Text blättern muss. Dadurch wird natürlich Platz eingespart, weshalb die Applets und die Beschreibungen *gleichzeitig* angezeigt werden können. Die Begleittexte auf einer Seite in einem anderen Browserfenster anzuzeigen wäre nicht sehr benutzerfreundlich, da der Benutzer in diesem Fall häufig zwischen den beiden Fenstern wechseln müsste. Die didaktischen Aspekte dieser Methode werden in Abschnitt 4.2 behandelt.

## 2.3 Weg der Daten

Es mag beim Begutachten des Quelltextes der Dateien auffallen, dass sehr häufig Daten bzw. Werte zuerst in das `menu`-, `menubottom`- oder `hidden`-Frame geleitet und von dort aus „weiterverarbeitet“ werden. Dies mag zunächst umständlich erscheinen, denn man könnte mit JavaScript die Daten auch direkt an das jeweilige Ziel transportieren. Dabei würde jedoch das Problem auftreten, dass beim Neuladen der aktuellen HTML-Seite bzw. beim Laden einer anderen neuen HTML-Seite an Stelle der aktuellen, allen Variablen wieder die Initialisierungswerte zugewiesen werden würden, wodurch alle vom Benutzer gelieferten Werte verloren wären. Die Frames `menu`, `menubottom` und `hidden` sind absolut statisch, d.h. beim Starten des Framesets wird ein bestimmtes HTML-Dokument in eines dieser Frames geladen und dieses HTML-Dokument bleibt immer in diesem Frame, wird also nie neu geladen oder durch ein anderes HTML-Dokument ersetzt.

# 3 Benutzerführung

## 3.1 Begleittexte

Bei den Begleittexten wurde sehr viel Wert auf gute Übersichtlichkeit und leichte Verständlichkeit gelegt. Die Texte enthalten viele Bilder zum besseren Verständnis des jeweils behandelten Themas und es wird auf viele Praxis-Beispiele eingegangen, um beim Benutzer Interesse zu wecken und um dem Benutzer den „Sinn und Zweck“ des Themas näher zu bringen. Zudem wurde versucht, die benötigten Formeln möglichst übersichtlich und klar strukturiert darzustellen, sodass der Inhalt der jeweiligen Formel auch nachvollziehbar ist, wenn der Benutzer den Text um die Formel herum noch nicht gelesen hat<sup>11</sup>.

## 3.2 Navigation und Orientierung

Zur Navigation dient zum einen ein Java-Menü<sup>12</sup> am linken Rand des Start-Framesets (siehe Abschnitt 2.1), das alle Seiten des Projektes systematisch sortiert auflistet, zum anderen existieren in jedem HTML-Dokument sehr viele Links zu verwandten Themen, wodurch der Benutzer schnell zu den Seiten des Projektes findet, die ihn interessieren. In dem gelb-grünen Frame oben werden der Pfad der jeweiligen HTML-Seite und – wenn der Benutzer mit der Maus über einen Link fährt – eine kurze Beschreibung des jeweiligen Links angezeigt.

---

<sup>11</sup>Dies wird dadurch erreicht, dass alle Variablen tabellarisch aufgelistet und in Worten erklärt sind. Ein Beispiel ist im Java-Menü unter „Begleittexte – Brechungsgesetz“ zu finden.

<sup>12</sup>Dieses Java-Menü wurde nicht von mir, sondern von einer außenstehenden Person programmiert.

### 3.3 Fehlermanagement

Fehlermanagement spielt bei den Applets eine große Rolle, weil die Applets ungeheuer flexibel sind und somit viele Sonderfälle vom Benutzer eingestellt werden können.

Zunächst wurde beim Programmieren der Applets darauf geachtet, dass bei allen einstellbaren Sonderfällen die Ausgaben des Applets korrekt sind und keine chaotischen Konstruktionen angezeigt werden<sup>13</sup>. Als nächstes wurden Hinweismeldungen programmiert, die den Benutzer auf den jeweiligen Sonderfall hinweisen. Diese Hinweise werden mit JavaScript gesteuert und erscheinen im Applet-Frameset im unteren Frame in der untersten Zeile, also unter dem Begleittext (siehe Abschnitt 2.2).

Es kann jedoch außer den Fehlern im Applet selbst auch zu Fehlern des Applets allgemein kommen. Wenn der Benutzer beispielsweise die in Abschnitt 1.5 erwähnten Systemanforderungen nicht erfüllt, wird der Browser das Applet nicht anzeigen können und eine Fehlermeldung ausgeben. Ähnliches würde passieren, wenn der Benutzer hinter einer Firewall sitzt, die mit Java zu penibel umgeht. Es wird daran gearbeitet, auch für diese Fehler eigene Hinweismeldungen zu programmieren, die den Benutzer darauf hinweisen, warum das Applet nicht angezeigt wird und welche Maßnahmen dagegen unternommen werden können.

## 4 Didaktische Aspekte

### 4.1 Ersatz für reale Experimente

Es ist zwar allgemein bekannt, dass es in der Schulphysik von immens großer Bedeutung ist, das Interesse des Schülers zu wecken, und in Physikbüchern steht auch oft „Probiere dieses Experiment doch selbst einmal aus!“, jedoch nimmt sich ein Schüler nur äußerst selten die Zeit, das Experiment durchzuführen, weil der Aufbau viel zu lange dauern würde und ihn das Ergebnis „ja sowieso nicht interessiert“.

Genau hier setzen die Java-Applets dieser Arbeit an. Wenn ein Applet geladen ist, kann der Schüler ganz einfach die Maus nehmen und das Experiment am Bildschirm durchführen. Dies geschieht sehr schnell und ohne großen Aufwand, weshalb die Wahrscheinlichkeit groß ist, dass man den Schüler dazu bringen kann weiterzublättern und immer neue interessante Sachen auszuprobieren.

### 4.2 Begleittexte

Bei den Begleittexten wurde u.a. darauf geachtet, dass mit den leichten, angenehmen, bekannten, etc. Dingen nicht übertrieben wird, da sich der Schüler sonst vermutlich – wie bei einem Computerspiel bei dem er *immer* gewinnt – langweilen würde. Genauso wurde aber auch versucht, das Schwierige, Unangenehme, Unbekannte, etc. in Grenzen zu halten, weil der Schüler bei Überforderung ganz einfach das Browserfenster schließen und aufgeben würde. Aus diesen beiden Gründen wurde sehr viel Wert darauf gelegt, die teilweise komplizierten und schwer verständlichen Zusammenhänge der geometrischen Optik in der *richtigen Mischung* aus leicht/schwierig, angenehm/unangenehm, bekannt/unbekannt etc. zu vermitteln.

---

<sup>13</sup>Teilweise wurden die Koordinatenwerte von ziehbaren Punkte auch so limitiert, dass bestimmte Sonderfälle nicht mehr einstellbar sind.

Einen weiteren wichtigen didaktischen Aspekt der Begleittexte stellt zudem die Zerstückelung der Texte unterhalb der Applets in zweizeilige Abschnitte dar (siehe Kapitel 2.2). Gerade für Schüler ist diese Lösung sehr gut geeignet, denn die Wahrscheinlichkeit, dass ein Schüler einen Text mit 50 Zeilen Physik freiwillig liest, ist zugegebenermaßen sehr gering. Dagegen ist es schon gut möglich, dass ein Schüler einmal eben zwei Zeilen liest, die zunächst überhaupt nichts mit Physik zu tun haben, sondern die insgeheim versuchen, den Schüler zum Weiterblättern „einzufangen“. In den ersten Zeilen wird versucht, Interesse, Lust oder sogar Faszination beim Schüler zu wecken. Mit der Zeit wird dann ganz nebenbei auch ein wenig auf die physikalischen Grundlagen eingegangen, wobei der Schüler in regelmäßigen Intervallen zum *interaktiven Selbstaussprobieren* im Applet aufgefordert wird. Es wird also nicht versucht, den Schüler direkt zu belehren, sondern vielmehr wird versucht, das Interesse bzw. die Lust des Schülers zu wecken. Wenn nämlich die Lust auf ein bestimmtes Thema bei einem Schüler erst einmal geweckt ist, ist es sehr gut möglich, dass dieser Schüler den Rest von ganz alleine erledigt.

### 4.3 Überwachung

Obwohl eine Überwachung<sup>14</sup> des Schülers mit JavaScript sehr leicht realisierbar wäre, wurde absichtlich darauf verzichtet, weil auf den Schüler kein Druck ausgeübt werden soll. Es ist sehr wahrscheinlich, dass der Schüler, wenn er „nicht gleich weiter weiß“, einfach das Fenster schließt und aufgibt. Dies soll durch „Freiheit“ (d.h. der Schüler kann jederzeit weiterblättern) innerhalb der Benutzerführung verhindert werden.

Lediglich Aufgaben zu den Applets werden überwacht bzw. geprüft, wobei diese Aufgaben aber auf jeden Fall in einem anderen neuen Begleittext im unteren Frame angezeigt werden. So kann der Schüler nach wie vor die eigentliche Benutzerführung „frei“ durchlaufen, hat aber *zusätzlich* die Möglichkeit, Aufgaben zum jeweiligen Thema mit Antwortüberprüfung zu bearbeiten.

## 5 Ähnliche Produkte

Applets, die mit den Applets dieser Arbeit vergleichbar wären, sind im Internet meines Wissens nach nicht zu finden. Alle mir bekannten Applets zur geometrischen Optik besitzen zum einen eine unzureichende Benutzerfreundlichkeit (z.B. erfolgt das Verändern der Position eines Punktes ausschließlich durch Schieberegler), zum anderen sind sie meist sehr oberflächlich, d.h. sie simulieren nur primitive Konstruktionen mit Näherungen, während auf viele Gesetzmäßigkeiten, die bereits in der Mittelstufe deutscher Gymnasien behandelt werden sollen, verzichtet wird.

So finden sich zu der einfachen Näherung des paraxialen Strahlenganges dünner Linsen  $\frac{1}{f} = \frac{1}{g} + \frac{1}{b}$  zwar einige gelungene Applets im Internet, jedoch wagen sich selbst „Java-Physik-Profis“ kaum an die Simulation von ganzen Systemen dünner Linsen bzw. sogar dicker Linsen, also z.B. von Teleskopen, Mikroskopen oder Fotoapparaten. Das liegt daran, dass ein Java-Programmierer jeden einzelnen Konstruktionsschritt mit aufwendigen algebraischen

---

<sup>14</sup>Gemeint ist, dass der Schüler den Text nicht sofort weiterblättern kann, sondern dass überwacht wird, ob er die geforderten Anweisungen bereits ausgeführt hat. Erst nach positiver Überprüfung wird dem Schüler die Möglichkeit zum Weiterblättern gegeben.

Formeln berechnen muss, was gerade bei komplexen Systemen kaum überschaubar ist. Geometria/Geoscript nimmt dem Programmierer diese lästigen Aufgaben ab und ermöglicht somit sehr komplizierte Konstruktionen, die trotzdem von Benutzern ohne jede Vorkenntnis bedient werden können. Zudem hat Geometria einige sehr nützliche Features (siehe Abschnitt 1.2).

Es sei auch angemerkt, dass große Optikfirmen selbstverständlich über Software verfügen, die wesentlich komplexere optische Systeme berechnen und simulieren, als es die Applets dieser Arbeit vermögen. Die Applets dieser Arbeit sind jedoch nicht dazu gedacht, mit solcher Profi-Software mithalten bzw. sogar konkurrieren zu können. Vielmehr sollen sie Schülern, Studenten und Interessierten die Grundprinzipien der geometrischen Optik möglichst interaktiv und anschaulich vermitteln und Fortgeschrittenen die Möglichkeit zur schnellen Berechnung optischer Zusammenhänge geben.

## 6 Danksagung

Mein Dank gilt allen, die mich bei diesem Projekt unterstützt haben. Ganz besonders danke ich Dr. Maxim Darscht für die Erklärungen zur Optik, Dr. Timo Ehmke für die Geometria-Hilfe, Holger Heidrich für die Verbesserungsvorschläge, Martin Kamp für die Unterstützung und Erklärungen, Peter Kraemer für die Unterstützung, Ideen und Erklärungen, dem Rotary-Club Nürnberg-Erlangen für das Sponsoring des Computers und den Mitarbeitern der Universitäten Würzburg und Jena für die Erklärungen und Kopien. Nicht zuletzt möchte ich mich bei Christoph Groth für das Korrekturlesen und den Tipp, GNU Emacs und  $\text{\LaTeX}$  zu verwenden, bedanken.

Marcel Schmittfull  
(marcel-sl@gmx.de)

Für den Klaus Tschira Preis für Jugendsoftware  
Juni 2002

## A Aktueller Entwicklungsstatus

Zum jetzigen Zeitpunkt (15. Juni 2002) befindet sich das Projekt noch in der Entwicklungsphase. Es ist erst eines von ca. 15 vorgesehenen Applets vollständig realisiert, das Totalreflexion-Applet. In den nächsten Wochen bzw. Monaten sollen jedoch die restlichen Applets (voraussichtlich Ebener Spiegel, Billard, Hohlspiegel, Parabolspiegel, Lichtwellenleiter, Planparallele Platte, Brechung an Kugeloberfläche, Dünne Linse, Dicke Linse (Hauptebenen), Lochkamera, Auge, Lupe, Galilei-Fernrohr, Kepler-Fernrohr, Mikroskop und Schmidtspiegel) fertig gestellt werden (vgl. Anhang D). Auch die HTML-Seiten sind größtenteils noch im Aufbau.

Das Projektes ist zur Zeit ausschließlich für die Klaus-Tschira-Wettbewerb-Juroren zum Download verfügbar, an einer öffentlichen Verfügbarkeit des gesamten Projektes über das Internet wird jedoch gearbeitet. Aktuelle Informationen sollten unter den folgenden Internetadressen abrufbar sein:

<http://home.arcor.de/martin-sl/tschira/download.html>

<http://www.schulphysik.de/java-optik>

<http://www.schulphysik.de/japtik>

## B Dateistruktur

Die Startseite befindet sich in der Datei `index.html` im Wurzelverzeichnis des Projektes. Von dieser Seite aus lassen sich alle HTML-Dokumente und Java-Applets laden.

Es wurde folgende Ordnerstruktur verwendet:

- Allgemeines<sup>15</sup> im Wurzelverzeichnis des Projektes,
- Applets in einem Unterordner des Ordners `ap`, wobei der Unterordner ein Kürzel für das jeweilige Applet darstellt<sup>16</sup>,
- Begleittexte im Ordner `be`,
- Bilder im Ordner `images`.

## C Geometria/Geoscript

Die Programmiersprache Geoscript an sich ist recht simpel: In einer Datei mit der Endung `.style` werden Werte gespeichert, die das Aussehen des Applets beeinflussen, z.B. die Gesamtgröße des Applets und die Standardfarben der Punkte und Linien. Die eigentliche Konstruktionsbeschreibung wird in einer Datei mit der Endung `.script` festgelegt. In jedem einzelnen Konstruktionsschritt wird dabei ein Objekt definiert:

```
e[1] = P1; point; draggable; 5.0,6.0;
e[2] = P2; point; fixed; 2.0,2.0; „hidden„
e[3] = s1; line; connect; P1,P2;
```

<sup>15</sup>Zum Beispiel zentrale Layout- und Design-Dateien.

<sup>16</sup>Beispielsweise `tore` für „Totalreflexion“, insgesamt also `ap/tore`.

Der Objektbezeichner (`P1`) wird durch einen Klassenbezeichner (`point`), einen Unterklassenbezeichner (`dragable`), Objektdaten (5.0,6.0 – Koordinaten) und evtl. Layout-Angaben („`hidden`“) definiert. Die beiden Dateien werden von einer HTML-Seite mit dem `<applet>` Tag aufgerufen. Zusätzlich kann das Applet mit JavaScript-Funktionen kommunizieren und Bilder einbinden (siehe Abschnitt 1.2).

Das Programmieren in dieser Sprache lässt sich mit dem Editor „GNU Emacs“ erheblich vereinfachen. Ich habe mit der Programmiersprache LISP einen eigenen Modus für Geometria programmiert, wodurch z.B. statt „`point; dragable`“ einfach „`p dr`“ geschrieben werden kann. Die Nummerierungen `e[n]` wurden durch eine weitere LISP-Funktion automatisch durchgeführt.

Eine Benutzeranleitung und eine Konstruktionsreferenz zu Geometria befinden sich im Internet unter <http://www.geometria.de/>.

## D Jugend forscht – Schüler experimentieren

Im Frühjahr 2002 habe ich bereits Applets zur geometrischen Optik bei dem Wettbewerb „Jugend forscht – Schüler experimentieren“ eingereicht und wurde Landessieger Bayern in der Kategorie Physik. Unter <http://www.schulphysik.de/japtik/> sind alle Applets und Begleittexte meiner damaligen Arbeit frei verfügbar.

Für den Klaus Tschira Preis für Jugend-Software habe ich die Applets und die Begleittexte noch einmal vollkommen neu als *eigenstehende Arbeit* geschrieben und sehr viele wesentliche Verbesserungen vorgenommen.

Marcel Schmittfull  
(marcel-sl@gmx.de)

Für den Klaus Tschira Preis für Jugendsoftware  
Juni 2002