Desenvolvimento de ferramentas no iGeom: utilizando a geometria dinâmica no ensino presencial e a distância

SEIJI ISOTANI [†]

Orientador:

LEÔNIDAS DE OLIVEIRA BRANDÃO

DISSERTAÇÃO APRESENTADA AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA UNIVERSIDADE DE SÃO PAULO
COMO PARTE DOS REQUISITOS PARA OBTENÇÃO DO
TÍTULO DE MESTRE EM CIÊNCIA DA COMPUTAÇÃO

São Paulo
Abril de 2005

[†]Durante a elaboração deste trabalho o autor contou com apoio financeiro do CNPq.

Desenvolvimento de ferramentas no iGeom: utilizando a geometria dinâmica no ensino presencial e a distância

Este exemplar corresponde à redação final
da dissertação devidamente corrigida e
defendida por Seiji Isotani
e aprovada pela comissão julgadora.

São Paulo, 1 de Abril de 2005.

Área de Concentração: Ciência da Computação

Banca examinadora:

- Prof. Dr. Leônidas de Oliveira Brandão - IME-USP
- Profa. Dra. Leliane Nunes de Barros - IME-USP
- Prof. Dr. Franck Bellemain - CAC-UFPE

Agradecimentos

Meus sinceros agradecimentos ao professor *Leônidas de Oliveira Brandão*, pela forma dedicada e paciente com que me orientou. Em todos os momentos esteve sempre disposto e pronto a ajudar.

Aos membros da minha banca de Mestrado *Leliane* e *Franck*.

À *Anarosa* pela ajuda na preparação dos slides da apresentação.

À todos os professores e alunos envolvidos com o projeto iMática, iGeom e SAW.

Aos meus pais, *Sadao* e *Naoko*, pela presença em todas as etapas de minha vida. Agradeço por todo o esforço, carinho, dedicação e confiança que depositaram em mim. Ambos são pessoas exemplares. Sem a ajuda de vocês não seria possível finalizar esta dissertação.

Aos meus irmãos, *Shigueo* e *Mina*, pelo apoio e companhia, dividindo brigas, desafios e alegrias. Vocês me ajudaram a amadurecer e me tornar uma pessoa melhor. Um obrigado especial ao Shigueo por ser companheiro de faculdade e me ajudar durante todo o período de adaptação no meu primeiro ano de curso no IME-USP.

À minha namorada *Monike*, por seu amor, carinho e paciência que me ajudaram a manter a tranquilidade durante esses dois anos de pós-graduação.

Aos meus amigos do IME-USP e ao grupo de teatro quIMEra.

Aos meus grandes amigos de faculdade *Hideo* e *Rafael* pela excelente companhia desde o início da graduação até os dias de hoje.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro (processo 132928/2003-5) durante o desenvolvimento desta dissertação.

Resumo

Neste trabalho, apresentamos o desenvolvimento de ferramentas no programa **iGeom - Geometria Interativa na Internet**, para ensino-aprendizagem de Geometria, dando destaque aos recursos que facilitam a integração e uso deste programa, principalmente em ambientes de educação a distância via Internet.

Atualmente, este tipo de programa é bastante conhecido e a Geometria que ele possibilita é usualmente denominada **Geometria Dinâmica**. Em poucas palavras, um programa de Geometria Dinâmica é a implementação computacional da régua e do compasso, permitindo que os objetos construídos sejam movidos mantendo-se às propriedades da construção.

Dentre os principais recursos desenvolvidos, destacamos a autoria e a validação automática de exercícios e a comunicação com servidores, que podem ser utilizados para integrarem o iGeom em sistemas gerenciadores de cursos pela Web. Deste modo, se integrado a um sistema gerenciador, estes recursos podem ser utilizados para facilitar a tarefa do professor, que poderá criar exercícios diretamente pela Web e não precisará avaliar pessoalmente as respostas de cada aluno, e também para que o aluno saiba de imediato se sua solução está dentro do esperado pelo professor.

Abstract

In this work, we present the development of tools on the software **iGeom - Interactive Geometry on Internet**, which can be used for teaching and learning Geometry. We highlight the features that simplify the integration and the use of the iGeom in distance education through Internet mainly.

Presently, this kind of program is well known and the Geometry it allows is usually named **Dynamic Geometry**. Concisely, a Dynamic Geometry software is the computational implementation of the ruler and compass. It allows the user to drag objects and to automatically redraw the whole construction while preserving their mathematical properties.

We focus three features among the main developed ones: the communication, the authoring and the automatic checking of exercises. These resources simplify the teacher's task, when integrated into a learning management system. With them, the teacher can create exercises directly on the Web leaving to the software the task of checking each student's answer. Moreover, the student can have, in real time, an evaluation of his own performance.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ATP	Automatic Theorem Proving
ACT	Atomic Component of Thought
CERN	European Center for Nuclear Research
EAD	Educação a Distância
GD	Geometria Dinâmica
HTML	Hypertext Markup Language
IME	Instituto de Matemática e Estatística
ITC	Instructional Technology Council
JVM	Java Virtual Machine
NCSA	National Center for Supercomputing Applications
PC	Personal Computer
POLI	Escola Politécnica
USP	Universidade de São Paulo
UFRJ	Universidade Federal do Rio de Janeiro
WWW	World Wide Web

Capítulo 1

Introdução

O computador digital tem sido empregado no ensino, praticamente, desde seu surgimento em 1945[1], mas é a partir da década de 80, com o aparecimento dos computadores pessoais (*personal computers*), os PCs, que a utilização desta máquina e de seus recursos (*software*) provocaram grande impacto na educação (Oldknow, 1997; Kortenkamp, 2001).

No momento, tanto no Brasil quanto em outras partes no mundo, ocorre uma grande expansão do uso da Internet no ensino e, consequentemente, a demanda por pesquisas nesta área têm aumentado consideravelmente (Litto et al., 2004).

Através dos avanços da Internet e das ferramentas de suporte a educação a distância, tornou-se possível difundir o conhecimento de forma extremamente rápida e atender às demandas por cursos com flexibilidade de horário e local. Neste contexto, os *ambientes virtuais* se transformam em salas de aula, onde alunos e professores se comunicam e interagem através de recursos como *chats*, fóruns de discussão, e-mails e lousas virtuais, dentre outras ferramentas colaborativas.

A utilização das novas tecnologias, principalmente as de comunicação e de interação, vem reforçando a reestruturação do método tradicional de ensino, denominado por (Freire, 1987) de "concepção bancária da educação". Nesta concepção, o professor é a figura central do aprendizado, cabendo ao aluno assimilar, de forma passiva e sem considerar o seu ritmo de aprendizagem, todo o conteúdo exposto no quadro-negro. Em oposição a este método tradicional, alguns pesquisadores e educadores defendem a educação "problematizadora" onde o aluno aprende através das situações-problema expostas pelo professor (Clements, 2000; Marrades & Gutiér-

[1]Para saber mais sobre a história dos computador visite a linha do tempo computing history http://www.hofstra.edu/ComputingHistory/.

rez, 2000).

Um matemático que se dedicou também ao ensino, Pólya (1978), defende que os métodos educacionais devem privilegiar o desenvolvimento das habilidades e técnicas matemáticas através da resolução de exercícios. Para Pólya, uma boa educação começa quando o próprio aluno descobre por si só a solução de problemas.

Além da resolução de problemas, um ponto importante no processo de aprendizagem é fornecer rapidamente ao aluno a validação das soluções dos exercícios/problemas realizados por ele, seja no modo presencial ou à distância. Como muitos trabalhos observam, a falta de uma avaliação/validação imediata dificulta a aprendizagem e pode causar a desmotivação do aluno (Hara & Kling, 1999; Kirby, 1999; Hentea et al., 2003).

Outra característica importante no processo de aprendizagem é o desenvolvimento, por parte do professor, de material adequado para atender às necessidades do curso e dos alunos. Em cursos à distância, as características mais desejadas no desenvolvimento e apresentação do material, segundo Jones (1996) e Hentea et al. (2003), são: interface simples e adaptável, autonomia, flexibilidade, interatividade e organização.

1.1 Delimitação do Problema

O uso do computador pode trazer grandes benefícios ao ensino de Matemática, mas para isso é necessário escolher programas adequados e uma metodologia que tire proveito das características positivas do computador, como boas representações gráficas e rapidez em cálculos. Um bom exemplo deste benefício é a Geometria Dinâmica, que resumidamente pode ser entendida como a geometria da régua e compasso implementada no computador.

A Geometria é, sob nosso ponto de vista, a área da Matemática que mais se beneficiou com o uso do computador e de suas tecnologias, quando se considera o ensino-aprendizagem. A razão de nossa crença pode ser ilustrada por um antigo ditado atribuído a Confúcio: *"O aluno ouve e esquece, vê e se lembra, mas só compreende quando faz"*. Ou seja, para aprender é necessário fazer e a Geometria Dinâmica auxilia o fazer, permitindo que o aluno vivencie situações-problema e descubra por si só, relações entre os objetos matemáticos (Brandão & Isotani, 2003).

O nome "**Geometria Dinâmica**" (GD) hoje é largamente utilizado para especificar a Geometria implementada em computador, a qual permite que objetos sejam movidos mantendo-se

todos os vínculos estabelecidos inicialmente na construção. Este nome pode ser melhor entendido como oposição à geometria tradicional de régua e compasso, que é "estática", pois após o aluno realizar uma construção, se ele desejar analisá-la com alguns dos objetos em outra disposição terá que construir um novo desenho.

A GD começou a ganhar destaque na década de 90 (Botana & Valcarce, 2002), principalmente com a popularização dos programas comerciais *Cabri Geometry* (Laborder & Bellemain, 1997) e *Geometer's Sketchpad* (Jackiw, 1995). Ambos foram tema de diversas pesquisas e trabalhos pedagógicos sobre o uso da GD no ensino.

O **iGeom - Geometria Interativa na Internet**, seguindo a linha dos programas citados anteriormente, é um programa de Geometria Dinâmica que começou a ser desenvolvido em 2000 no Instituto de Matemática e Estatística da Universidade de São Paulo (IME-USP).

Coordenado pelo professor Leônidas de Oliveira Brandão, um dos objetivos do desenvolvimento deste programa foi "democratizar" o uso da Geometria Dinâmica, permitindo que qualquer estudante ou professor, com acesso a um micro-computador, pudesse usufruir dos benefícios da GD.

Para isso, o programa iGeom é disponibilizado gratuitamente e é implementada na linguagem de programação **Java** (Java, 2004), por esta permitir grande portabilidade (possibilidade de uso em diferentes computadores/sistemas operacionais) e possibilitar seu uso diretamente em páginas Internet, na forma de ***applet*** (programa Java especialmente projetado para a Web).

Dentro deste contexto, o presente trabalho contribui para o desenvolvimento de novos recursos didáticos no programa iGeom, recursos estes que facilitam sua utilização efetiva em sala de aula e em ambientes de educação a distância.

O foco de nosso trabalho foi o aperfeiçoamento do iGeom de uma maneira geral e, em particular, a implementação de recursos de comunicação, publicação, autoria e validação automática de exercícios. Tais funcionalidades oferecem novas perspectivas para o uso de programas de GD pela Internet. Através delas tornou-se possível: (a) prover a integração de um programa de GD com um sistema gerenciador de cursos na Web; (b) validar de forma rápida e automática os exercícios realizados; (c) oferecer respostas rápidas ao aluno; (d) auxiliar o professor a criar/editar conteúdo para Internet e a verificar a produção de seus alunos, além de outras aplicações.

1.2 Justificativas

Apesar da existência de diversos trabalhos bem sucedidos na utilização dos programas de GD no ensino presencial como, por exemplo, o livro editado por King & Shattschneider (1997) e os trabalhos de Jones (2000), Marrades & Gutiérrez (2000) e Hollebrands (2003), notamos uma grande lacuna de trabalhos que tenham como objetivo utilizar a GD via Internet em cursos presenciais ou à distância. Isso porque são poucos os programas de GD que podem ser executados diretamente de uma página de Internet e mais raro ainda são os que possuem ferramentas de comunicação, produção e validação de conteúdo.

Se por um lado a interatividade e a dinâmica dos programas de GD se mostram um excelente recurso de aprendizagem para os alunos, os mecanismos de comunicação, autoria e validação são fundamentais para permitir que o professor consiga produzir material de boa qualidade e agilizar o processo de validação dos exercícios realizados.

Atualmente, desconsiderando o iGeom, conhecemos apenas dois programas de GD que possuem ferramentas de produção e validação de conteúdo para Web: o *Cinderella* (Kortenkamp, 1999) e o *C.a.R.* (Grothman, 1999). Porém, nenhum deles possui recursos de comunicação. Sem estes recursos, os programas citados não permitem a troca de informação com um servidor e, portanto, a sua utilização em cursos à distância torna-se restrita, principalmente quando queremos verificar ou analisar as interações de um aluno com o programa.

A proposta deste trabalho foi preencher esta lacuna, visando oferecer a oportunidade de professores e alunos usufruírem os benefícios da Geometria Dinâmica através de um conjunto de recursos no iGeom que facilitam sua imersão em ambientes presenciais e pela Internet.

1.3 Objetivos

O objetivo geral deste trabalho foi dotar (e testar) o iGeom de ferramentas facilitadoras ao ensino-aprendizagem de Geometria para uso em ambientes presenciais, semi-presenciais ou à distância.

Buscamos, por um lado, providenciar recursos que auxiliem o professor na produção de material didático e no acompanhamento de seus alunos, e por outro, trazer facilidades para um aluno adquirir os conhecimentos geométricos.

Dentre os recursos que desenvolvemos destacamos as ferramentas para autoria e validação

automática de exercícios, comunicação com o servidor e a integração do iGeom com um ambiente gerenciador de ensino a distância.

Por **autoria** entende-se aqui o conjunto de funcionalidades que viabilizam a produção de exercícios. São exemplos destas funcionalidades: interface (janela) para criação de exercícios, a flexibilização das ferramentas (botões) disponíveis para o usuário e mecanismos para gravar/intepretar exercícios, principalmente para páginas Web.

O termo **validação** , segundo o dicionário Houaiss[2], significa: (a) ato ou efeito de validar, de tornar ou declarar algo válido, legítimo; (b) teste que comprova a validade, a correção ou concordância com padrões; (c) confirmação. Este termo será aqui utilizado para descrever o processo de comparação entre soluções de um exercício: a solução do aluno é comparada com uma solução fornecida pelo professor (gabarito), utilizando-se um critério de "distância", para confirmar/comprovar a validade da mesma.

O recurso de **comunicação** é a funcionalidade que permite a troca de informações entre o iGeom e um servidor ligado à Web, através de um protocolo padrão na Internet. Com este mecanismo podemos, por exemplo, enviar a solução do aluno para um exercício, além de outras informações sobre a interação do aluno com o iGeom.

Estes novos recursos já foram testados no ambiente **SAW - Sistema de Aprendizagem pela Web**, um sistema para gerenciamento de cursos na Web, que está sendo desenvolvido por outro estudante de mestrado do IME-USP (Moura & Brandão, 2004). Através da integração iGeom+SAW realizamos alguns testes em cursos semi-presenciais[3]. A realização destes testes foram de grande importância, para identificar as diversas dificuldades do professor e do aluno com a inserção desta tecnologia no ensino de Geometria e mostrar o quanto facilita o trabalho do professor.

1.4 Organização do Texto

Este texto está estrutura em 6 capítulos. O capítulo 2 apresenta o conceito de Geometria Dinâmica, alguns dos benefícios para o professor e para o aluno, um exemplo de aplicação pouco explorado nos programas de GD e alguns dos principais programas disponíveis atualmente.

[2]Dicionário Eletrônico Houaiss - http://www.dicionariohouaiss.com.br/

[3]Cursos realizados com aulas presenciais, em laboratórios, e também a distância, via Internet.

O Capítulo 3 apresenta o programa de GD iGeom, contanto um pouco sobre seu desenvolvimento e alguns de seus principais recursos disponíveis.

O Capítulo 4 discute o papel da GD na educação a distância (EAD) e como os novos recursos desenvolvidos no iGeom podem contribuir para o uso deste programa em ambientes de EAD.

O Capitulo 5 faz uma breve introdução aos recursos de autoria e validação automática de exercícios presentes nos programas de GD, iGeom, Cinderella e C.a.R., e apresentar alguns aspectos da técnica utilizada para fazer a validação automática nestes programas e como esta foi implementada no programa iGeom.

Por último, o Capítulo 6 contém algumas conclusões sobre esta dissertação, discute os resultados alcançados e propostas para futuras implementações ou extensões.

Capítulo 2

A Geometria Dinâmica

> *"Dynamic is the opposite of static. Dynamic also connotes action, energy, even hype. Dynamic geometry is active, exploratory geometry carried out with interactive computer software."*
>
> (King & Shattschneider, 1997)

Como citado anteriormente, podemos entender por Geometria Dinâmica (GD) a implementação computacional da "geometria tradicional", aquela de régua e compasso. O termo "dinâmico" do nome pode ser melhor entendido como oposição à estrutura "estática" das construções da geometria tradicional. Na GD, após o aluno realizar uma construção, ele pode alterar as posições dos objetos iniciais e o programa redesenha a construção, preservando as propriedades originais.

Em função desta possibilidade de alterar objetos preservando-se a construção, podemos dizer que a GD é uma geometria do tipo 1-construção, N-testes, enquanto a tradicional de régua e compasso é do tipo 1-construção, 1-teste (Brandão, 2004). Deste modo, um programa de GD possibilita, a partir de uma única construção, efetuar um número arbitrário de testes, o que seria praticamente impossível com régua e compasso.

Um exemplo simples que pode ilustrar o "dinamismo" desta geometria é a construção da mediatriz de dois pontos dados, A e B (exemplo 2.1). Para construir a mediatriz basta encontrarmos dois pontos distintos, que eqüidistem de A e de B, e por eles traçar a reta resposta r (mediatriz). Uma vez efetuada a construção podemos mover os pontos A ou B pela área de desenho e o programa que implementa a GD, automaticamente, redesenhará todos os objetos preservando suas propriedades. Desta forma, a reta r continuará visualmente sendo a mediatriz de A e B (Figura 2.2).

Exemplo 2.1 *Dados dois pontos, A e B, construir sua mediatriz (lugar geométrico dos pontos que equidistam dos dois pontos dados).*

Construção 2.2 *Dados: ponto A, ponto B (resposta: reta r)*

1. *construir a circunferência $C0$, com centro no ponto A e contendo B,*
2. *construir a circunferência $C1$, com centro no ponto B e contendo A*
3. *construir a reta r definida pelos pontos C e D, interseções entre $C0$ e $C1$.*

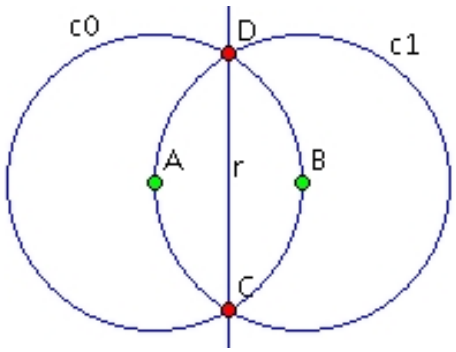

Figura 2.1: Exemplo de construção da mediatriz

Outras duas características interessantes nos programas de GD são: introduzir a necessidade de melhor formalização das construções e, devido ao dinamismo, facilitar a verificação de validade da mesma. A formalização é necessária devido ao computador não admitir ambiguidades[1] e utilizando o dinamismo podemos verificar mais facilmente se a construção preserva as propriedades esperadas. No exemplo 2.1, se o usuário não construiu corretamente a mediatriz, ao arrastarmos o ponto A (ou B) pela área de desenho aparecerá algum erro que pode ser identificado visualmente.

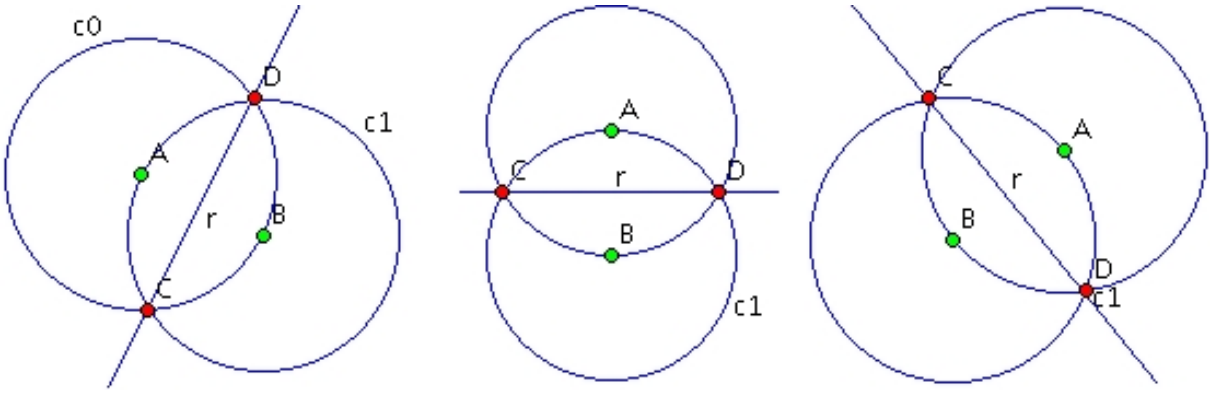

Figura 2.2: Exemplo da mediatriz em diversas configurações

[1]Um exemplo de ambiguidade seria "construir uma circunferência definida pelos pontos A e B", enquanto uma versão não ambígua poderia ser "construir a circunferência de centro A, que passa pelo ponto B".

2.1 Benefícios da GD

A discussão sobre vantagens/desvantagens pedagógicas entre as duas formas de se “fazer” geometria pode ser conduzida sob diferentes pontos de vista. Neste trabalho, abordaremos apenas a interatividade e visualização como mecanismos facilitadores da aprendizagem na GD.

O uso da GD no ensino da Geometria traz boas possibilidades de mudança em uma área que vem sendo negligenciada no ensino. Segundo Gravina (1996) e Usiskin (1987), o ensino da Geometria recebe pouca atenção, tanto no ensino fundamental e médio, quanto no ensino superior. Além disso, frenqüentemente a geometria é ensinada de forma mecânica, sem a preocupação em destacar os conceitos envolvidos (Crowley, 1987).

De um lado, notamos problemas na forma tradicional de se ensinar Geometria. Como nota Gravina:

> *“Os livros escolares iniciam o ensino de Geometria com definições, nem sempre claras, acompanhadas de desenhos bem particulares, os ditos desenhos prototípicos. Por exemplo, quadrados com lados paralelos às bordas da folha de papel, retângulos sempre com dois lados diferentes, altura em triângulos sempre acutângulos, entre outros. Isto leva os alunos a não reconhecerem desenhos destes mesmos objetos quando em outra situação. E mais, os alunos passam a acreditar que a posição relativa do desenho ou seu traçado particular façam parte das características do objeto, o que os leva a estabelecer desequilíbrios na formação dos conceitos. O aspecto de construção de objetos geométricos raramente é abordado. Dificilmente encontramos no livro escolar a instrução "construa", e no entanto, esta é uma das atividades que leva o aluno ao domínio de conceitos geométricos.”*
>
> (Gravina, 1996)

Por outro lado, temos o potencial interativo e aberto de um programa de GD que, segundo Arcavi & Hadas (2000), podem ser comparados a laboratórios virtuais nos quais os estudantes podem manipular, investigar e aprender matemática. Como observa Marrades & Gutiérrez (2000), discutindo o aprendizado de geometria através dos programas de GD:

> *"A contribuição dos programas de Geometria Dinâmica segue em dois ramos. Primeiro, provêem um ambiente no qual estudantes podem experimentar livremente. Dessa forma, eles podem facilmente verificar suas intuições e conjecturas durante o processo de procura de padrões, propriedades, etc. Segundo, estes programas provêem formas não tradicionais para os estudantes aprenderem e entenderem os métodos e conceitos matemáticos ... permitindo construir figuras complexas e facilmente realizar, em tempo real, uma quantidade enorme de transformações nestas figuras, proporcionando ao estudante o acesso a uma grande variedade de exemplos que dificilmente seriam possíveis em ambientes não computacionais ou em ambientes computacionais estáticos."*
>
> (Marrades & Gutiérrez, 2000)

Do ponto de vista do aprendizado, também podemos notar vantagens da GD sobre a geometria estática. Usando o modelo de aprendizado de Geometria proposto pelos van Hiele (Crowley, 1987)[2], que classificam os níveis cognitivos de aprendizado de Geometria em cinco (visualização, análise, dedução informal, dedução formal e rigor), notamos que a Geometria Dinâmica pode ser bem empregada nos três primeiros níveis. Nestes níveis iniciais, o estudante está começando a abstrair os conceitos matemáticos e, deste modo, a experimentação pode contribuir muito.

No ensino tradicional, o aluno apenas "ouve", não sendo incentivado a ter uma postura investigativa (ativa) e nem sendo desafiado a construir seu próprio conhecimento. Em uma aula de Geometria tradicional o professor enuncia conceitos, definições e propriedades que, muitas vezes, são apenas memorizados e futuramente reproduzidos pelo aluno sem sua devida compreensão.

Segundo Gravina (1996) e Arcavi & Hadas (2000), a GD proporciona uma nova abordagem ao aprendizado geométrico, onde conjecturas são feitas a partir da experimentação e criação de objetos geométricos. Deste modo, podemos introduzir o conceito matemático dos objetos a partir da resposta gráfica oferecida pelo programa de GD, surgindo naturalmente daí o processo de argumentação e dedução.

Para King & Shattschneider (1997), destacam-se como principais benefícios e aplicações de um sistema computacional de Geometria Dinâmica: a prova de teoremas, a precisão e visualização, as explorações e descobertas, as transformações e lugares geométricos e, por fim,

[2]Este modelo é anterior ao surgimento da GD, que aparece por volta de 1987, enquanto o trabalho principal dos van Hiele é da década de 50 (Lindquist & Shulte, 1987).

a simulação de micromundos. Vale a pena destacar os principais argumentos dos autores em relação a cada uma delas:

- **Prova de teoremas.** Embora a Geometria Dinâmica não possa provar teoremas, a capacidade de experimentação de hipóteses pode motivar a busca pela prova de um teorema, pois induz à convicção de sua validade. Da mesma forma, pode ajudar e sugerir caminhos para a prova formal.

- **Precisão e visualização.** A construção da geometria é feita pelo estabelecimento de relações geométricas entre os elementos (perpendicularismo, paralelismo, pertinência, ângulo, etc). Pode-se medir ângulos e distâncias e calcular relações com precisão, permitindo facilmente a verificação empírica de hipóteses e teoremas. Os conceitos de um teorema podem ser compreendidos por visualização. Adicionalmente, a precisão também é importante porque construções imprecisas podem conduzir o aluno a conclusões errôneas.

- **Exploração e descoberta.** A manipulação de construções permite que se explore a Geometria e que "novas" relações e propriedades sejam descobertas. Muitas vezes, os próprios alunos "redescobrem" teoremas em sala de aula.

- **Transformações e lugares geométricos.** Pela sua capacidade de realizar transformações em figuras geométricas, programas de Geometria Dinâmica são ideais para o estudo de isometrias, similaridades e outras funções. Animando figuras e traçando lugares geométricos de pontos pré-definidos, estes aplicativos também podem explicitar problemas e propriedades normalmente não abordadas na literatura por sua inerente dificuldade.

- **Simulação e micromundos.** Indo muito além da abstração da Geometria, as simulações que podem ser construídas com programas de Geometria Dinâmica permitem ilustrar conceitos de cinemática e óptica, entre outros. Por outro lado, oferecem também a possibilidade de criação de micromundos geométricos, a exemplo daqueles concebidos no âmbito da linguagem Logo (Papert, 1999), que propõe ao aluno um campo de experimentação onde ele constrói o conhecimento através da manipulação dos objetos (Bellemain, 2002). Neles, o aluno pode vivenciar experiências geométricas, algumas pré-concebidas pelo professor e muitas outras descobertas ao acaso, através da exploração interativa e de sua criatividade.

Vale observar que o comentário de King & Shattschneider (1997) sobre "prova de teoremas" na GD, pode começar a ser explorado com alunos que estejam no segundo nível de compreensão, proposto por van Hiele, a análise. Quando o aluno está neste nível, começa a analisar os conceitos geométricos envolvidos, por exemplo, podendo usar a experimentação para discernir características dos objetos (um exemplo simples de experimentação no segundo nível é o aluno perceber que um quadrado também é um retângulo, por conter todas as propriedades exigidas deste último, ou perceber quando um retângulo torna-se um quadrado).

Ainda sobre a "prova de teoremas", podemos acrescentar outra razão para o uso da GD: os contra-exemplos. Com a GD o aluno pode mais facilmente encontrar uma configuração que sirva de contra-exemplo a uma conjectura em estudo. Esta observação também pode servir como resposta a uma das críticas mais comuns contra a GD: *a visualização dispensaria, ou desestimularia, a necessidade de prova matemática*. Esta crítica pode ser encontrada, por exemplo, em Munzner (1996).

Em resumo, como a GD possibilita visualizar uma mesma construção de diversas formas, e assim facilitar a compreensão do comportamento geométrico dos elementos envolvidos, podemos utilizar um programa de GD para revelar relações geométricas intrínsecas que poderiam passar desapercebidas numa representação estática (Rodrigues, 2002). Com isso, o professor pode incentivar o espírito investigativo do aluno, solicitando ao final uma justificativa para as relações encontradas, ou seja, a prova matemática.

2.1.1 O Professor

De acordo com Freire (1987), o professor deve livrar-se do estigma de detentor do conhecimento e se transformar em um guia que oferece dicas e estímulos para que os alunos aprendam. Nesta abordagem de ensino, o professor será o "parceiro" do aluno liderando atividades que visem a exploração e a descoberta, e que favoreçam a criatividade e a interação do aluno com o assunto abordado.

Neste processo de ensino-aprendizagem, o professor irá incentivar e ajudar o aluno a descobrir por si só o mundo matemático, seus conceitos e suas propriedades. As dicas e conselhos do professor, devem ser tomados como valiosos preceitos que servirão como guias durante o processo de descoberta. Dessa forma, é possível estimular a curiosidade sobre a matemática, e não apenas incentivar a busca por uma resposta.

O uso do computador nas escolas vem recebendo grande atenção por parte dos educadores.

Clements (2000) acredita que o uso desta tecnologia trás grandes benefícios ao ensino, não apenas pelas inovações nas formas de se apresentar o conteúdo, mas também por causa da inevitável mudança nos métodos de ensino que favorece a visão de parceria e troca de experiências entre professores e alunos.

Em diversas escolas do ensino médio e superior o uso do computador está cada vez mais presente no cotidiano e vêem sendo incorporado ao currículo escolar, principalmente na área de Matemática (Hollebrands, 2003). Devido a este fato, o professor tem sido progressivamente cobrado a utilizar o computador em suas aulas.

No ensino de Matemática, os programas de GD podem ajudar o professor a introduzir os conceitos de matemática/geometria utilizando o computador. Além disso, a forma como será apresentado o conteúdo poderá proporcionar um maior aprendizado por parte dos alunos. Pois como destaca Arcavi & Hadas (2000), as atividades com a GD oferecem ao professor ferramentas para trabalhar com as capacidades de visualizar, transformar, generalizar, refletir e se comunicar com a informação, habilidades consideradas fundamentais para guiar o aluno durante uma atividade matemática. Além disso, segundo estes mesmos autores o uso sistemático dos programas de GD proporciona: (a) atividades que sejam interligadas por seus conteúdos e suas diferentes representações; (b) a oportunidades para o aluno pensar e questionar as atividades, propondo e respondendo questões (as respostas não precisam estar corretas); e (c) com as respostas dos alunos é possível que o professor realize uma reflexão das atividades para chegar em uma conclusão formal.

Segundo Marrades & Gutiérrez (2000):

> *"Os programas de Geometria Dinâmica auxiliam o professor a criar ambientes de aprendizado nos quais o aluno pode experimentar e observar a permanência ou não de propriedades matemáticas, propondo e verificando conjecturas de forma muito mais simples se comparada a qualquer outra forma tradicional utilizando régua e compasso.*"

Contudo, todo o potencial benéfico que os programas de GD oferecem requer o preparo adequado por parte do professor e um grande esforço de sua parte na preparação de conteúdo. Como muito bem observa Bellemain (2002), a tarefa de utilizar os programas de GD não é simples, pois embora os programas de GD permitam elaborar situações que favorecem a construção de conhecimentos, eles não ensinam nada. Cabe ao professor criar bons problemas que usem recursos da GD, propiciando ao aluno o aprimoramento das suas habilidades matemáticas/geométricas.

Uma outra questão que surge com a inserção dos programas de GD (e de outros programas para ensino por computador) é a dificuldade, por parte do professor, em validar a resposta do aluno durante a realização de um exercício e acompanhá-lo durante as atividades propostas (Bellemain, 2002). A razão disso é que a maior parte da interação ocorre entre aluno-computador e o professor na maioria das vezes não têm acesso a essa interação.

Neste contexto, as ferramentas que facilitam a produção, validação e gerenciamento de exercícios são fundamentais para auxiliar o professor na elaboração e análise das atividades com os programas de GD. Segundo Clements (2000), os programas que permitem ao professor desenvolver exercícios práticos, que podem ser resolvidos utilizando diferentes estratégias, são os mais recomendados para o ensino de Matemática. Além disso, através das ferramentas de validação automática de exercícios, diminuímos a carga de trabalho do professor e, se estas ferramentas estiverem interligadas a um sistema de gerenciamento de conteúdo/curso (por exemplo, o já citado SAW - seção 4.3.2), podemos catalogar os trabalhos realizados pelos alunos. Posteriormente, o professor pode usar este catálogo para verificar as estratégias e as dificuldades encontradas por seus alunos durante as atividades realizadas, tanto no modo presencial quanto à distância.

2.1.2 O Aluno

Fazendo uso dos benefícios da GD citados anteriormente, tornamos o aluno a peça-chave de seu próprio aprendizado. Contudo, neste paradigma de ensino-aprendizagem, a postura participativa do aluno é requisito mínimo para a compreensão do assunto em discussão. Dessa forma, um aluno motivado a aprender pode fazer uso dos programas de GD em prol de um aprendizado mais ativo onde a busca pelo conhecimento pode produzir resultados sensivelmente positivos (Hollebrands, 2003).

A transição do método tradicional de ensino para o ensino auxiliado por computador pode afetar tanto o professor quanto o aluno. Assim como o professor já habituado ao ensino tradicional precisa adaptar-se aos recursos computacionais, o mesmo pode ocorrer com o aluno que precisará adaptar-se aos sistemas de ensino por computador e abandonar o comportamento passivo.

Neste contexto, após o período de adaptação, o aluno precisa estar constantemente buscando desafios e, sempre que possível, compartilhando suas dúvidas e experiências com os colegas e professores. É através desta interação, e da superação das dificuldades encontradas durante as

atividades propostas pelo professor, que o aluno atinge a maturidade para compreender o conteúdo apresentado. Para Santos & Sola (2001), as atividades baseadas na resolução de exercícios são as mais importantes no ensino de Geometria, pois ajudam o aluno a fazer e testar conjecturas e, dessa forma, adquirir o conhecimento necessário para entender os conceitos e aplicá-los posteriormente.

Com a ajuda dos programas de GD, podemos criar exercícios interativos nos quais, se o aluno agir ativamente modificando as características de vários objetos matemáticos, ele aprenderá pesquisando, analisando e verificando o que ocorre genericamente (Melo et al., 2000). Nesta abordagem é possível disponibilizar representações gráficas de objetos geométricos que aproximam o objeto representado na tela do computador (desenho) ao objeto teórico (figura/conceito), favorecendo o desenvolvimento de uma leitura geométrica dos desenhos e contornando, assim, uma das grandes dificuldades no ensino da Geometria (Bellemain, 2001).

Segundo o estudo realizado por Hannafin (2001), reportando a inserção de um programa de GD em salas de aula do ensino médio, após o período de adaptação e reconhecimento do programa os alunos sentiram-se mais livres (tanto para perguntar, quanto para fazer conjecturas), trabalharam mais e demonstraram maior interesse no assunto abordado.

Além desses benefícios, os recursos computacionais de validação automática e gerenciamento de exercícios podem auxiliar o aprendizado do aluno. Segundo Hentea et al. (2003), após a realização de cada exercício, o oferecimento de respostas rápidas ao aluno contribui para o seu aprendizado. Uma das razões para esta afirmação é devido ao aluno poder tirar as dúvidas imediatamente após o surgimento das mesmas. Além disso, utilizando uma ferramenta de gerenciamento podemos: (a) armazenar as construções de cada aluno para seu estudo posterior; e (b) oferecer exercícios de acordo com a velocidade do aluno em resolvê-los, respeitando seu ritmo de aprendizagem.

2.2 Exemplo de aplicação

Um exemplo pouco explorado de aplicação utilizando os programas de GD é a introdução dos conceitos de algoritmo. Isso ocorre porque poucos são os programas de GD que dispõe de recursos para agrupar passos de construção na forma de uma função geométrica (que chamaremos de *"script"*).

Informalmente, podemos dizer que um **algoritmo** é uma seqüência finita de passos que

aplicada a um conjunto de **dados de entrada** produz um conjunto de **dados de saída** (ou resposta). Além disso, a menos de uma classe particular de algoritmos, um algoritmo deve ser **determinístico**, ou seja, sempre que for aplicado sobre um mesmo conjunto de entradas, deve produzir o mesmo conjunto de saídas. Observe que, se o conjunto de entrada for vazio, a saída do algoritmo será sempre a mesma.

Uma característica importante de um algoritmo é que ele resolve uma **classe de problemas** e não uma **instância**. Por exemplo, um algoritmo de ordenação para N números inteiros (digamos com $N < 10^8$), ordena qualquer conjunto com até N inteiros, em qualquer configuração (isto é, qualquer que seja a permutação, dentre as $N!$ possíveis). A aplicação do algoritmo sobre um particular conjunto de inteiros, constitui a resolução de uma instância do problema.

Esta observação permite entendermos melhor a diferença entre a GD, que é do tipo 1-N e a geometria estática, tipo 1-1: uma solução geométrica implementada em GD, na prática constitui um algoritmo, enquanto a correspondente solução estática equivale a uma aplicação do algoritmo geométrico sobre um conjunto fixado de dados (estáticos e, portanto, único).

A caracterização de soluções geométricas como algoritmos, pode ser melhor percebida utilizando-se a GD, pois ao finalizar uma construção e testá-la com outras configurações (de entrada), fica claro que a mesma pode ser aplicada a qualquer outro conjunto de entradas (dentre os possíveis). Por outro lado, como observa Rodrigues (2002), as soluções obtidas pela geometria da régua e compasso são estáticas e particulares, pois não podem ser alteradas e nenhuma delas garante o significado genérico de sua definição (o desenho de um círculo, por exemplo, possui um centro e raio, ambos fixos, mas o conceito de círculo não depende de valores arbitrários).

Para introduzir o conceito de algoritmo geométrico, vamos examinar o exemplo 2.1, sobre a construção da mediatriz de dois pontos dados, A e B. Na tabela 2.1, apresentamos um esquema dos passos para obter a mediatriz r de A e B.

Tabela 2.1: Um algoritmo para construção da mediatriz

Mediatriz(A**,**B**)**:
  $C0$:= Circ(A,B);
  $C1$:= Circ(B,A);
  ln := Intersec($C0$,$C1$,n); //interseção Norte
  ls := Intersec($C0$,$C1$,s); //interseção Sul
  r := Reta(ln,ls);
  Resposta r;

Note que os passos descritos podem ser aplicados a quaisquer pares de pontos (não coin-

cidentes) constituindo assim um algoritmo. Os comandos $Circ$, $Intersec$ e $Reta$ podem ser entendidos como primitivas (funções) geométricas: $Circ(X, Y)$ é a circunferência centrada em X e passando pelo ponto Y; $Intersec(X, Y, p)$ é um dos pontos de interseção entre os objetos X e Y (podem existir dois, neste caso definidos por "norte", se $p = n$, ou "sul", se $p = s$); e $Reta(X, Y)$ é a reta que contém os pontos X e Y (se $X = Y$, a reta não será única).

Deste modo, é natural esperar que programas de GD permitam que construções geométricas sejam armazenadas explicitamente na forma de **funções**, como o fazem o iGeom, GSP e Cabri. Estas funções geométricas, por agruparem seqüências de comandos, recebem o nome de **scripts** (como nas versões atuais do iGeom e do GSP) ou **macros** (como no Cabri). Uma vez armazenada uma função, pode-se: (a) marcar os objetos de entrada (na ordem correta) e depois selecionar a função desejada (como no iGeom e GSP); ou (b) selecionar a função e depois marcar os objetos de entrada (Cabri). Como discutiremos posteriormente na seção 2.3.1, a primeira forma pode ser dita do tipo “seleção+ação” enquanto a segunda é do tipo “ação+seleção”.

A possibilidade de armazenar algoritmos geométricos como funções é bastante útil do ponto de vista didático, pois permite que o aluno armazene em funções as construções que utiliza mais freqüentemente e com isso possa concentrar sua energia nas tarefas novas. Os programas já citados (iGeom, Cabri e GSP) permitem até que uma função, já armazenada, seja invocada durante a geração de uma nova função e com isso aproveitamos funções prontas para construir outras mais complexas.

Exemplo nos quais os usos das funções geométricas são muito úteis são aqueles com processo de repetição, nos quais é necessário aplicar várias vezes a mesma seqüência de passos, como no exemplo 2.3, apresentado por Brandão (2002).

Exemplo 2.3 Aquiles e a Tartaruga (ou "paradoxo de Zenão").
Aquiles e uma tartaruga apostam uma corrida, sendo que Aquiles tem o dobro da velocidade da tartaruga, e por isso, a tartaruga larga à frente.

Para simular geometricamente este exemplo (Figura 2.3), definimos uma função *dist_metade* que obtém a próxima posição da tartaruga em relação a sua posição atual e a posição de aquiles (tabela 2.2).

Utilizando a função *dist_metade*, podemos simular esta hipotética corrida a partir de eventos discretos, definindo duas sequências, uma de tempo $\{t_k\}_{k\in\mathbb{N}}$ e uma de posição $\{P_k\}_{k\in\mathbb{N}}$. Sendo t_0 o instante de largada e P_0 a posição inicial da tartaruga, podemos definir as sequências, para

Tabela 2.2: Construção de uma função para o exemplo 2.3

dist_metade(A, B)
Entrada: ponto A, ponto B.
Saída: ponto F tal que F pertence semi-reta(A, B) e d(B, F)=d(A, B)/2.
1. r := semi_reta(A, B); semi-reta começando em A, passando por B
2. $C0$:= circ(B, A); circunferência centrada em B, passando por A
3. $C1$:= circ(A, B); circunferência centrada em A, passando por B
4. C := inters($C0, C1, n$); C é interseção "superior"(norte) entre as circunferências C0 e C1
5. D := inters($C0, C1, s$); D é interseção "inferior"(sul) entre as circunferências C0 e C1
6. $S1$:= segm(C, D); S1 é o segmento ligando C à D
7. E := inters($r, S1$); E é interseção entre r e S1
8. $C2$:= circ(B, E); circunferência centrada em B, passando por E
9. F := inters($C2, r, n$); F é interseção “superior” (norte) entre C2 e r

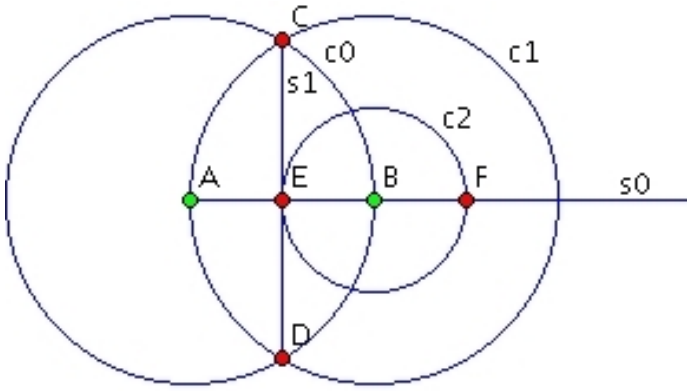

Figura 2.3: Representação gráfica para a função *dist_metade*

$k > 0$, da seguinte forma:

$$\begin{array}{ll} t_k: & \text{o instante em que Aquiles atinge a posição } P_{k-1} \\ P_k: & \text{a posição ocupada pela tartaruga no instante } t_k. \end{array}$$

Deste modo, a construção apresentada na Figura 2.3 serve para gerar os pontos P_k, como esquematizado na tabela 2.3.

Zenão argumentava que a metade de um número positivo (distância) é um número positivo e, deste modo, sendo o tempo e o espaço contínuos, Aquiles jamais alcançaria a tartaruga[3]. Como observado em Ávila (1999), o aparente paradoxo divulgado por Zenão ilustra a dificuldade que os matemáticos tinham, antes do surgimento do Cálculo Diferencial e Integral, com o conceito

[3]Note que o problema deste argumento é supor que a soma de infinitas parcelas positivas será sempre $+\infty$, o que não é verdade. No exemplo, temos uma soma de p.g. infinita, de razão menor que 1, $1/2+(1/2)^2+(1/2)^3+\cdots = 1$.

Tabela 2.3: Esquema para geração de respostas para o exemplo 2.3

Sendo A e B as posições iniciais, respectivamente, de Aquiles e da tartaruga, então:

1. $P_0 = B$ é a posição inicial da tartaruga e A é a posição inicial de Aquiles;
2. $P_1 \leftarrow dist\_metade(A, P_0)$ quando Aquiles chegar à posição P_0 a tartaruga estará em P_1 $\left(\text{da construção, } P_1 \in A\vec{P}_0 \text{ e } d(P_0, P_1) = \frac{d(A,B)}{2}\right)$;
3. $P_2 \leftarrow dist\_metade(P_0, P_1)$ quando Aquiles chegar à posição P_1 a tartaruga estará em P_2 $\left(P_2 \in P_0\vec{P}_1 \text{ e } d(P_1, P_2) = \frac{d(P_0,P_1)}{2} = \frac{d(A,B)}{2^2}\right)$

   e assim por diante.

infinito e infinitésimo.

Note que o algoritmo geométrico apresentado na tabela 2.3 possui um bloco de repetição (ou laço[4]), comum nas linguagens de programação usuais (como C, Pascal ou Java). O passo geral do algoritmo da tabela 2.3 é:

$$P_{k+1} \leftarrow dist\_metade(P_{k-1}, P_k), \text{ para } k > 1, \text{ sendo } P_0 = B \text{ e } P_1 = dist\_metade(A, B).$$

Assim, para efetuar a simulação utilizando *dist_metade*$(.,.)$ é necessário aplicá-la seguidas vezes. Entretanto, se na própria definição deste algoritmo incorporarmos uma chamada recorrente, a própria recorrência controla as múltiplas aplicações. Portanto, é natural imaginar que tal recurso também possa ser incorporado aos *scripts* na GD. A partir do *script dist_metade* podemos produzir um novo *script*, de nome *aquiles*, anotando a repetição através de uma **recorrência** (ou **recursão**). Usando a função *dist_metade*, o *script aquiles* é definido da seguinte forma:

Construção 2.4 $aquiles(A, B)$*: ponto* A*, ponto* B

1. $F \leftarrow dist\_metade(A, B)$ *de acordo com a construção da tabela 2.2*
2. $G \leftarrow aquiles(B, F)$ *recorrência, aplicada aos pontos* B *e* F

A recorrência é caracterizada pela instrução na linha 2. Entretanto existe um problema com o pseudo-código da construção 2.4 que é não indicar uma condição de parada. Nas linguagem de programação usuais, como C ou Pascal, utilizam-se um condicional para invocar a recorrência (e portanto, a recorrência é interrompida quando certa condição não é satisfeita). Em nosso caso, fazemos o controle definindo a priori o número de chamadas recorrentes. O número de vezes (conhecido como **profundidade**) que a recorrência será aplicada, , é definido na chamada da

[4]Sequência de passos repetitivos, que constitui a parte central do algoritmo.

função pelo usuário. Por exemplo, a Figura 2.4 mostra o uso do *script aquiles* com profundidade dois (poderia-se produzir

uma construção mais “limpa” examinando-se os objetos intermediários).

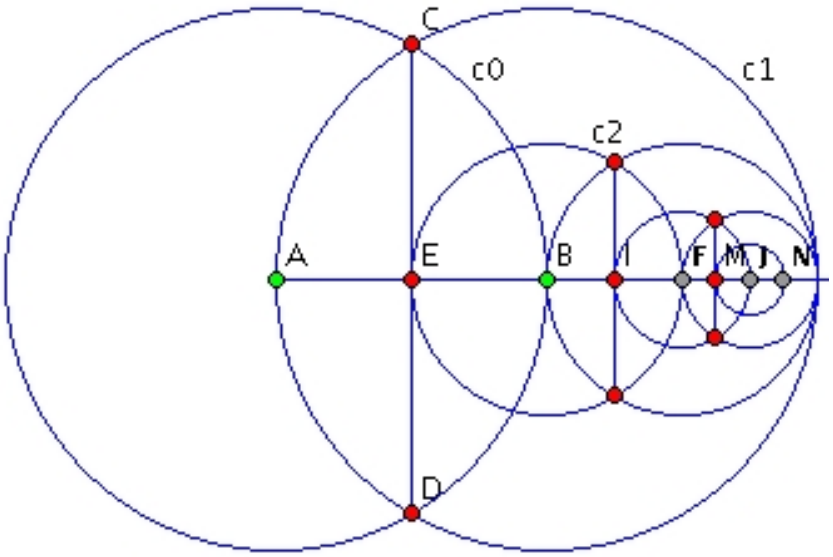

Figura 2.4: Aplicação do *script* recursivo *aquiles* com profundidade 2. Aplicações para os pares (A,B) = F, (B,F) = j, (F,j) = N.

Atualmente, conhecemos apenas dois programas de GD que permitem ***scripts* recorrentes**, o iGeom e o GSP. O uso de recorrência em *scripts*, nos programas de GD, permite uma elegante introdução ao conceito de algoritmo, sem a necessidade de explicar variáveis ou comandos do tipo *“for”*. Além disso, o uso de recorrência agiliza a construção de **fractais**[5] geométricos (Mandelbrot, 1983; Kappraff, 1991), com os quais podemos explorar conceitos de progressões geométricas, somatórios e até de convergência. O exemplo 2.3, da corrida entre Aquiles e a tartaruga, pode ser utilizado para iniciar tal atividade, seguindo-se de algum exemplo graficamente mais interessante, como o fractal baseado em circunferências apresentado na Figura 2.5.

[5]O nome fractal é devido à Benoit Mandelbrot, derivada do latim *fractus*, que significa quebrado, partido ou irregular. Apesar de não existir um definição universalmente aceita para fractal, uma característica comumente aceita é a **auto-similaridade**, que significa que podemos reconhecer o todo da figura olhando apenas uma parte da mesma.

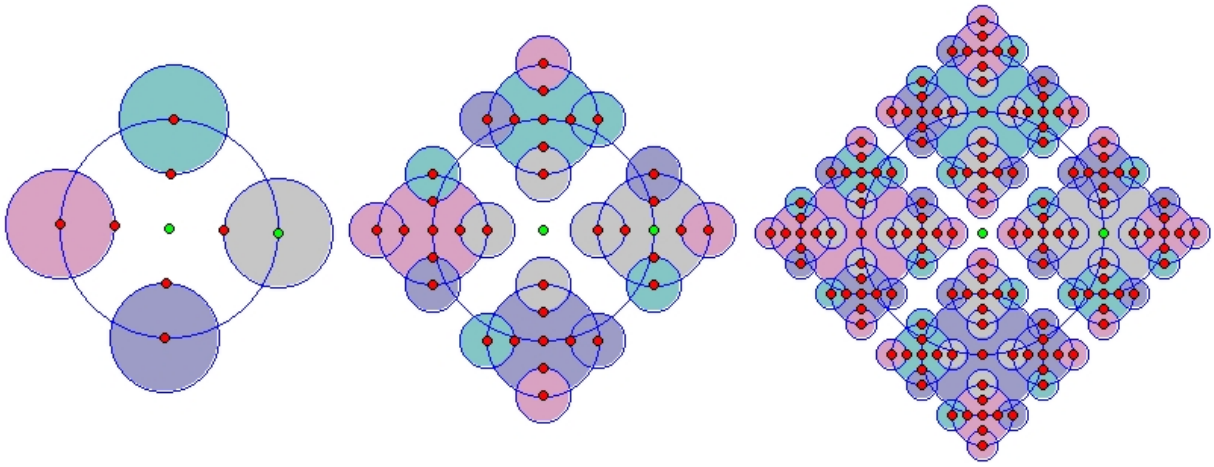

Figura 2.5: Exemplo de fractal baseado em circunferências gerado no iGeom

2.3 Alguns Trabalhos Relacionados

A Geometria Dinâmica (GD) começou a ganhar destaque na década de 90 (Botana & Valcarce, 2002), principalmente com a popularização dos programas comerciais *Cabri Geometry* (Laborder & Bellemain, 1997), que surgiu por volta de 1988, e *Geometer's Sketchpad* (GSP) (Jackiw, 1995), que apareceu pela primeira vez em 1989.

Devido a esta popularização, diversos trabalhos foram desenvolvidos para discutir e analisar os muitos aspectos pedagógicos do uso dos programas de GD na educação. Dentre estes trabalhos, Jones (2000) afirma que os temas principais dizem respeito a utilização da GD para o aprimoramento do raciocínio dedutivo e para o aprendizado baseado em experiências.

Para Kortenkamp (1999) e Botana & Valcarce (2002), o uso destes programas permitiu uma nova proposta de ensino de Geometria que visa explorar os mesmos conceitos da Geometria tradicional, mas através de atividades graficamente mais interessantes e interativas. King & Shattschneider (1997) acreditam que a GD se tornou uma ferramenta indispensável tanto para matemáticos quanto para cientistas preocupados com o ensino de matemática.

Segundo o trabalho de Kortenkamp (1999), em 1999 já existiam mais de 40 programas de GD. Contudo, eram poucos os programas portáveis, que podiam ser executados em diferentes computadores, com diferentes sistemas operacionais, ou que permitiam sua utilização diretamente em páginas Web. Atualmente, apenas os programas desenvolvidos em Java possuem todas estas características. Dentre os programas de GD implementados em Java destacamos: o Cinderella (Kortenkamp, 1999), o C.a.R. (Grothman, 1999), o Tabulae (Guimarães et al., 2002)

e o iGeom (Brandão & Isotani, 2003).

Como observa Oldknow (1997), todos os programas de GD apresentam muitas ferramentas em comum, dentre elas as de criação de objetos geométricos simples (como pontos, segmentos, retas, circunferências, entre outras); ferramentas para edição (mostrar/esconder objetos, rastrear ponto, gravar/recuperar construção); e ferramentas para construções clássicas (ponto médio, paralela, perpendicular, etc). Contudo, cada programa possui particularidades que os tornam singulares, por exemplo, na forma como ocorre a interação entre o usuário e o programa, ou no tipo de **plataforma** (computador + sistema operacional) que pode ser utilizado.

Nas subseções 2.3.3 até 2.3.7 apresentamos resumidamente cinco programas de GD (Cabri, GSP, Cinderella, C.a.R. e o Tabulae) e discutimos um pouco suas características e diferenças.

2.3.1 Questões de Interação

A interação entre o usuário e o programa de GD sempre foi foco de muitas discussões. Como observa Rodrigues (2002), em sua avaliação de interfaces de programas de GD, uma das questões mais discutidas no desenvolvimento destes programas é a forma como são efetuados os comandos para a criação e modificação dos objetos geométricos.

Na GD, objetos como retas ou circunferências, são construídos a partir de outros objetos. Por exemplo, pode-se construir uma circunferência $c0$ definida pelos pontos A e B, onde A é o centro da circunferência e o raio de $c0$ seja sempre a distância entre A e B. Podemos dizer que o recurso do programa de GD que permite construir um objeto, por exemplo, a circunferência $c0$, é uma **ação** e os pontos A e B são os objetos selecionados, sobre os quais a ação é aplicada para produzir $c0$.

Sobre este aspecto, existem duas **formas de interação** que podemos resumir em: escolher a ação antes (ação+seleção) ou depois (seleção+ação) dos objetos sobre os quais será aplicada. Segundo Rodrigues (2002), existe uma série de discussões sobre as vantagens e desvantagens de cada solução, sendo a determinação da melhor alternativa um recorrente motivo para pesquisas e debates. O principal motivo destas discussões é o antagonismo entre as implementações dos dois principais programas de GD do mercado, o Cabri, que implementa a ordem ação+seleção, e o GSP que implementa a seleção+ação.

Em geral, visando obter as vantagens de cada um dos dois formatos, os programas de GD lançados mais recentemente, como o Cinderella e o iGeom, adotam o formato livre onde as prin-

cipais ferramentas de construção funcionam tanto no formato ação+seleção quanto no formato seleção+ação. Esta abordagem permite que o usuário escolha qual é a maneira mais confortável (ou mais rápida) para construir, dependendo da situação.

Deste modo, os programas de GD podem ser classificados quanto à forma de interação em três tipos:

- pré-fixa (ação+seleção): seleciona primeiro a ação depois os objetos;
- pós-fixa (seleção+ação): seleciona primeiro os objetos e depois a ação;
- pré e pós fixa (formato livre): implementa ambas as opções.

2.3.2 A Linguagem Java

A linguagem **Java** originou-se como parte de um projeto de comércio eletrônico iniciado em 1991 e foi formalmente lançada em 1995 pela Sun Microsystems como uma linguagem de programação para Internet (Java, 2004). Esta linguagem foi concebida para o desenvolvimento orientado a objeto, porém, as principais diferenças em relação as outras linguagens de programação é o modo como os programas Java são compilados e executados.

Segundo Thomas et al. (1996), o aparecimento do Java redefiniu o conceito de desenvolvimento e distribuição de programas pois, enquanto os programas desenvolvidos em outras linguagens estão confinados a uma determinada plataforma, os desenvolvidos em Java são independentes de plataforma.

Ao contrário dos compiladores das linguagens como C ou Pascal, que produzem programas executáveis após a compilação, o compilador Java produz um arquivo semi-compilado que não pode ser executado diretamente pelo sistema operacional.

Os arquivos criados pelo compilador Java são interpretados por “máquinas virtuais”, conhecidas como ***Java Virtual Machine*** ou apenas **JVM**. A existência da JVM faz com que o programa desenvolvido em Java não tenha “contato” direto com o computador que o executa. Essa característica permite que os programas em Java sejam pré-compilados para rodar nesta máquina virtual e, dessa forma, qualquer computador “real” que possua uma JVM poderá executar os programas em Java.

O aparecimento da JVM e sua utilização pelos navegadores de Internet (como o Netscape, Mozilla, Internet Explorer, e muitos outros) permitiu a executação dos programas em Java di-

retamente em páginas Web. Estes programas em Java que podem ser interpretados pela Web recebem o nome de ***applet***.

A Figura 2.6 mostra como um *applet* é interpretado diretamente no computador-cliente através de um navegador Web. Segundo Thomas et al. (1996), antes do surgimento da linguagem Java, a maioria dos programas relacionados com a Web eram executados no servidor. Para este autor o surgimento do *applet* foi uma das grandes inovações na programação para a Web, pois além da portabilidade e da interatividade que estes programas oferecem, temos a possibilidade de trabalhar com imagens gráficas, permitido animações em tempo real diretamente de uma páginas Web. Por questões de segurança, os *applets* não têm acesso à memória ou aos arquivos do computador-cliente.

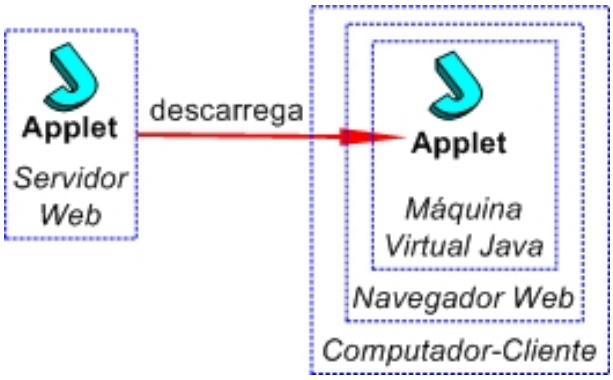

Figura 2.6: Execução dos *applets* através do navegador Web (Thomas et al., 1996)

Muitos programas de GD como o Cabri e o GSP foram desenvolvidos antes do surgimento desta linguagem, e portanto, não podem ser executados em qualquer plataforma nem diretamente pela Internet. Para resolver parte desta limitação, foram desenvolvidos interpretadores em Java (o CabriJava e o JavaSketchpad) que permitem que as construções gravadas nestes programas possam ser utilizadas para criar páginas interativas. Contudo, por razões comerciais óbvias, as funcionalidades destes programas na Web são limitadas, não permitindo ao usuário criar novos objetos ou fazer qualquer outro tipo de modificação. Em contrapartida, programas integralmente desenvolvidos em Java, como o iGeom, o C.a.R. e o Cinderella podem ser executados em qualquer plataforma e também via internet, permitindo que quase todos os recursos (a menos dos recursos de manipulação de arquivos) estejam disponíveis em páginas Web[6].

[6]Não conseguimos uma versão do Tabulae que funcionasse via Web.

2.3.3 Cabri Géomètre

O Cabri-Géomètre (Laborder & Bellemain, 1997) é o resultado das pesquisas conjuntas entre a Universidade Joseph Fourier de Grenoble e o Centro Nacional de Pesquisa Científica (CNRS) realizadas no Laboratório de estruturas discretas e didáticas, e posteriormente, seguida pelo laboratório Leibninz, ambos do IMAG - Institut d'Informatiqe et de Mathematiques Appliquees (Instituto de Informática e de Matemática Aplicada), sob a liderança de Jean-Marie Laborde. O estudo e desenvolvimento deste programa contou com a participação, principalmente, dos pesquisadores Franck Bellemain, Philippe Cayet e Yves Baulac.

A primeira versão do Cabri foi apresentado em 1988 no Congresso Internacional de Educação Matemática em Budapest. Atualmente o Cabri está em sua terceira versão (Cabri II Plus) e, embora o desenvolvimento continue sob os cuidados do IMAG, é comercializado pela Texas Intruments, tanto para computadores (PC's) quanto para certo modelos de calculadoras. Este programa é disponibilizados para PC's com sistemas operacionais Windows e MacOS (*Apple*).

O Cabri é um poderoso programa de GD que, além dos recursos usuais da GD (apresentados na seção 2.3), é dotado de diversas outras funcionalidades como a construção de cônicas, lugares geométricos, macros, animações, e muitas outras. Segundo Braviano & Rodrigues (2002), o Cabri é considerado o programa de GD mais utilizado nas escolas brasileiras.

2.3.4 Geometer Sketchpad - GSP

O GSP - Geometer Sketchpad (Jackiw, 1995) foi desenvolvido a partir de um projeto denominado "*Visual Geometry*" da Universidade de Swarthmore em meados de 1980. Coordenado pelos professores Eugene Klotz e Doris Schattschneider e implementado pelo aluno de graduação Nicholas Jackiw. Uma versão preliminar deste programa foi apresentada em 1989, enquanto a primeira versão comercial surgiu em abril de 1991, comercializada pela Key Curriculum Press.

O GSP está em sua quarta versão e, como o Cabri, pode ser executado em PC's que possuam sistema operacional Windows ou MacOS. Existem duas diferenças deste programa em relação ao Cabri que queremos destacar. A primeira delas a forma de interação do usuário com o programa durante a construção de objetos. Como discutido da seção 2.3.1, o Cabri utiliza-se da ordem pré-fixa (ação+seleção) enquanto o GSP faz uso da pós-fixa (seleção+ação). A segunda é a possibilidade de criar scripts (macros) recorrentes no GSP o que não é possível no Cabri.

2.3.5 Cinderella

O projeto Ciderella foi iniciado em 1992 por Jürgen Richter-Gebert e Henry Crapo sendo desenvolvido para a plataforma NeXT (uma plataforma proprietária que teve sua produção encerrada) até 1996, quando o pesquisador Ulrich Kortenkamp se juntou a equipe de desenvolvimento. Devido aos problemas de compatibilidade e portabilidade do NeXT, em 1996 o Cinderella foi totalmente reescrito utilizando a linguagem Java e, em 1999, teve sua primeira versão comercializada pela Springer-Verlag.

A versão atual do Cinderella é a 1.4, que apesar de ser muito semelhante aos seus antecessores, GSP e Cabri, apresenta algumas diferenças como: o suporte para múltiplas Geometrias (como euclideana, hiperbólica e outras) e a interação no formato livre (ação+seleção ou seleção+ação) na maioria das ferramentas. Contudo, a maior diferença é sua implementação em Java, o que permite usá-lo em qualquer plataforma e também diretamente em páginas Web. Além disso, um recurso pioneiro implementado neste programa, e que será discutido posteriormente no capítulo 5, é a autoria e a validação automática de exercícios.

2.3.6 C.a.R.

O nome do programa C.a.R. surgiu da tradução do alemão Z.u.L (Zirkel und Lineal) para o inglês Compass and Ruler (Compasso e Régua). O C.a.R. (Grothman, 1999) foi desenvolvido por René Grothman, professor da Universidade Católica de Eichstaett na Alemanha.

A primeira versão do programa C.a.R. foi desenvolvida para o computador Atari ST em 1988. Quatro anos mais tarde foi lançada a versão do C.a.R. para Windows. Contudo, com o surgimento da linguagem Java e sua posterior popularização, o professor René iniciou a reconstrução do C.a.R. na linguagem Java, ficando pronta em 1999.

A versão mais recente do programa C.a.R. é a 3.4 e é disponibilizada gratuitamente, inclusive o seu código fonte, através da Licença Pública Geral da GNU (GNU General Public License)[7]. Apesar de não dispor de tantos recursos (nem estabilidade) se comparado aos programas comerciais Cabri, GSP e Cinderella, o C.a.R. apresenta alguns recursos interessantes. Um recurso deste programa que daremos destaque é a autoria e a validação automática de exercícios (vide capítulo 5).

[7]Para obter mais informações sobre a GNU General Public License consulte o endereço: http://www.gnu.org/copyleft/gpl.html

2.3.7 Tabulae

O Tabulae (Guimarães et al., 2002) é um programa de GD desenvolvido no projeto Enibam (*http://www.tabulae.net*) do Instituto de Matemática da Universidade Federal do Rio de Janeiro (UFRJ), desde 1998. O desenvolvimento deste programa contou com a participação de professores e alunos de graduação e pós-graduação dos cursos de engenharia, matemática, informática e de desenho industrial da UFRJ.

Este programa é comercial[8], desenvolvido em Java e, assim como o C.a.R. e o Cinderella, pode ser executado diretamente em páginas Web e também permite a interação no formato livre.

[8] Até o presente momento, o Tabulae ainda não possui meios de distribuição ou venda, nem disponibiliza seus recursos na Web.

Capítulo 3

O Programa iGeom

> *"O iGeom: Geometria Interativa na Internet, proporciona recursos para facilitar o ensino e aprendizagem de Geometria, por um lado providenciando recursos que auxiliem o professor na produção de material didático e no acompanhamento de seus alunos, e por outro, trazendo facilidades para um aluno adquirir conhecimentos geométricos."*
>
> (Brandão & Isotani, 2003)

O programa de GD **iGeom** começou a ser desenvolvido no Instituto de Matemática e Estatística da Universidade de São Paulo (IME-USP) em 2000, coordenado pelo professor Leônidas de Oliveira Brandão. O objetivo inicial do projeto era disponibilizar um programa de GD gratuito e que pudesse ser utilizado via Web.

Com o intuito de facilitar a inserção do iGeom nos cursos de Geometria, o foco principal deste trabalho foi desenvolver e implementar novas funcionalidades (ferramentas) neste programa, de tal forma que fosse possível auxiliar tanto o professor em sua tarefa de criar exercícios e validar as respostas obtidas, quanto ao aluno oferecendo-lhe respostas rápidas para cada exercício realizado. Além disso, estes recursos precisavam funcionar como aplicativo e também via Internet (através do uso dos *applets*).

A primeira versão funcional do iGeom com autoria e validação automática de exercícios foi disponibilizada em março de 2004. Também no início deste mesmo ano, foi iniciado um projeto para o desenvolvimento de um sistema gerenciador de cursos à distância, denominado SAW - Sistema de Aprendizagem pela Web (Moura & Brandão, 2004), que pudesse receber módulos educacionais na forma de *applets*. O protótipo do SAW, testado ainda em 2004, fez uso dos recursos de comunicação, autoria e validação automática de exercícios do iGeom.

Atualmente, o iGeom (em conjunto com o SAW) é utilizado como principal ferramentas

na disciplina MAC118 - *Noções de Ensino de Matemática usando o Computador*, disciplina obrigatória para os alunos de Licenciatura em Matemática do IME-USP.

Neste capítulo, resumimos o histórico do desenvolvimento do iGeom e descrevemos algumas de suas principais funcionalidades, sem detalhes de implementação.

3.1 A História do iGeom

O **iGeom - Geometria Interativa na Internet** começou a ser desenvolvido em meados de 2000, em conjunto com o projeto **iMática** coordenado pelo professor Lêonidas de Oliveira Brandão do IME-USP. Este desenvolvimento começou quando eu e meu companheiro de faculdade Ricardo Hideo Sahara iniciamos nossos trabalhos de iniciação científica supervisionado por Brandão. Enquanto Sahara ficou responsável pela construção do programa iGeom (Sahara & Brandão, 2001), eu fiquei responsável pela reestruturação e algumas novas implementações do iMática (Isotani & Brandão, 2001). O projeto iMática, disponível no endereço *http://www.matematica.br*, tem como objetivo fornecer grande quantidade de conteúdo sobre Matemática e disponibilizar ferramentas gratuitas para o ensino via Web (por exemplo, o próprio iGeom).

A escolha da linguagem **Java** (vide seção 2.3.2) para implementar o iGeom foi para satisfazer a proposta inicial do iGeom de se tornar um programa de GD que funcionasse nas formas **aplicativo** e ***applet***. A versão "aplicativo" é a mais geral, permitindo gravar e recuperar arquivos nos vários formatos previstos, enquanto a versão "mini-aplicativo", ou *"applet"*, pode ser utilizada em qualquer navegador com interpretador Java.

A primeira versão funcional do iGeom implementada por Brandão e Sahara surgiu no início de 2001. Esta versão trazia os recursos básicos da GD e podia ser utilizado diretamente em páginas Web (Figura 3.1).

No primeiro semestre de 2001 ocorreu a parceria entre os professores Brandão e Eduardo Toledo dos Santos da Escola Politécnica da Universidade de São Paulo (POLI-USP) que também estava trabalhando com o desenvolvimento de programas de GD (Santos & Sola, 2001). Esta parceria consolidou-se através da orientação conjunta do trabalho de iniciação científica realizado por Fabiana Piesigilli (Piesigilli et al., 2002) que participou principalmente da primeira versão do gravador e interpretador de *scripts* recorrentes.

No início de 2003 iniciei meu projeto de mestrado no IME-USP sob a orientação de Brandão.

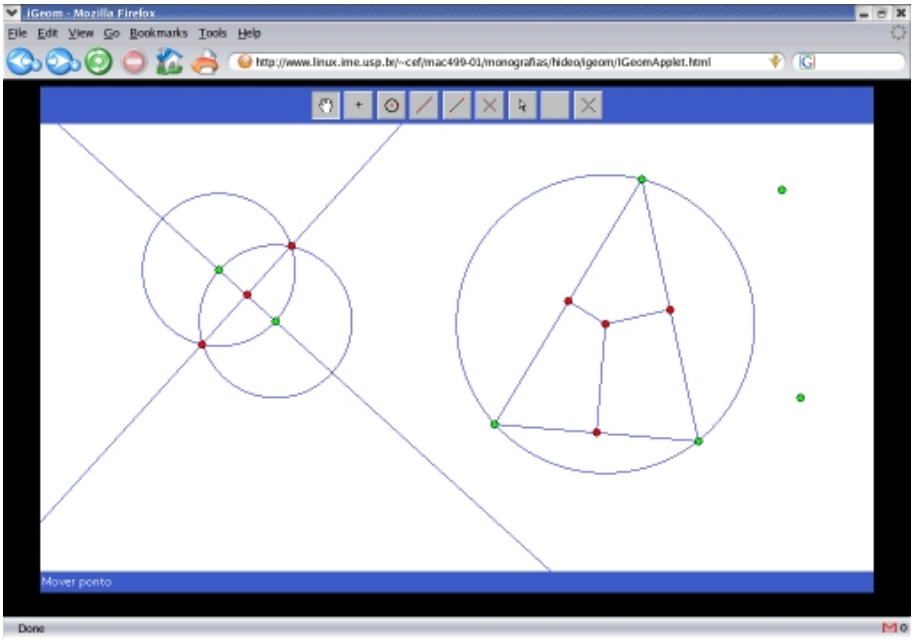

Figura 3.1: Versão do iGeom executada pela Internet em 2001

Nesta ocasião o iGeom estava em sua versão **2.0**, possuindo mais de 100 classes e milhares de linhas de código. Após o período de compreensão do código do iGeom, que durou aproximadamente 6 meses, iniciamos o período de desenvolvimento e reestruturação do programa. Dentre as principais contribuições (ferramentas implementadas) deste trabalho citamos a construção do sistema de autoria, validação automática de exercícios e do sistema de comunicação do iGeom com o servidor Web.

O iGeom atualmente está na sua versão **3.1** e é disponibilizado gratuitamente, seja para uso direto na Internet ou para ser descarregado a partir do *Site* iMática *http://www.matematica.br*. Lá também será encontrado um pequeno roteiro para o uso do programa e exemplos, incluindo cursos de Geometria produzidos por alunos de Licenciatura em Matemática do IME-USP.

3.2 A Interface Principal

A interface do iGeom, na versão aplicativo, é estruturada a partir das **opções de menus**, um **menu de botões**, dividido em **menu principal** e **secundário**, a **área de desenho** e uma **barra de mensagens**. Na Figura 3.2 apresentamos a atual interface principal do iGeom (versão **3.1**). A interface da versão *applet* é análoga, mas não dispõe das opções de menus.

Esta interface foi desenvolvida com o intuito que o iGeom pudesse ser utilizado nas formas aplicativa e *applet*. A menos das opções de gravação, que não são permitidas em *applet* (portanto via Internet) por razões de segurança, todas as demais estão acessíveis a partir dos botões. Além disso, estes botões foram construídos, assim como o restante do programa, usando uma quantidade pequena dos recursos disponíveis na atual versão do Java, de modo a ampliar as possibilidades de portabilidade. Explicando de modo mais técnico, o iGeom está escrito em Java puro, usando apenas a classe gráfica AWT e deste modo pode ser interpretado por qualquer navegador que tenha embutido o Java 1.1 ou superior (atualmente o Java está na versão 1.5).

A linguagem Java é bastante dinâmica, às vezes sofrendo grandes alterações. No início do desenvolvimento do iGeom surgiu a versão 2 do Java que possuia o pacote gráfico *Swing*. Apesar deste pacote trazer vários recursos simplificadores, optamos por não utilizá-lo para que o iGeom pudesse ser utilizado por um número maior de sistemas. Por exemplo, o Netscape versão 4.77, ainda comum naquele ano, trazia uma JVM embutida que não interpretava os comandos implementados com o *Swing*.

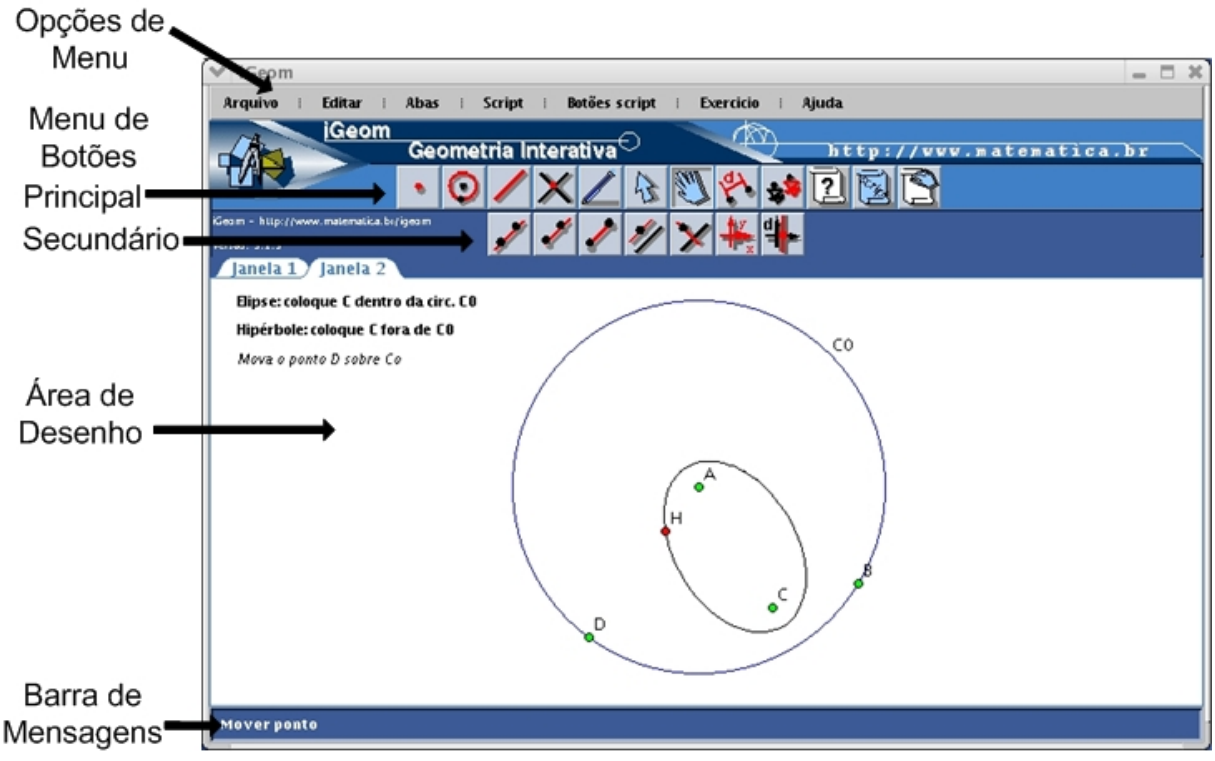

Figura 3.2: Janela Principal do iGeom

As **opções de menus** contém recursos para edição de arquivos (gravar/recuperar/exportar

construções), criação de *scripts* e exercícios, dentre outros.

Como dito anteriormente, no **menu de botões** disponibiliza-se todos os recursos do iGeom, excetuando-se aqueles relativos à gravação/leitura em arquivos. Este menu é dividida em dois níveis: na fileira superior está **menu principal** e na inferior o **menu secundário**. Um botão do menu principal pode ter ou não uma lista de botões associados no menu secundário. Caso tenha, este botão não terá qualquer ação correspondente, a não ser servir para abrir sua lista secundária.

Por exemplo, o botão de criação de isometrias no menu principal, ao ser clicado, fará aparecer no menu secundário as opções relacionadas com isometrias: translação, reflexão e rotação (Figura 3.3). O mesmo processo ocorre quando o botão de criação de circunferências é clicado, fazendo surgir no menu secundário as opções relacionadas com a criação de circunferência (Figura 3.4).

Figura 3.3: Opções de isometria

Figura 3.4: Opções de circunferência

A **área de desenho** funciona com uma folha de papel em branco, na qual é possível desenhar os objetos geométricos (pontos, retas, circunfências, etc) e interagir com eles. Nesta área, além dos objetos, podemos inserir textos e **cálculos dinâmicos** como, por exemplo, o enunciado de um exercício e a distância entre dois pontos. Em relação aos cálculos, estes são atualizados automaticamente, ou seja, quando calculamos a distância entre dois pontos A e B, ao movê-los pela área de desenho, o valor da distância é atualizado "instantaneamente" (gerando a impressão de dinamismo).

Finalmente, na parte inferior da interface está a **barra de mensagens** que irá apresentar informações sobre um botão (o que faz, como utilizá-lo, etc). Por exemplo, quando o cursor do mouse é posicionado sobre um botão é apresentada alguma informação sobre o mesmo. Assim, ao colocar o cursor sobre o botão de isometrias no menu principal surgirá a mensagem *"Isometrias (translação, reflexão e rotação): marque primeiro os objetos depois clique na isometria desejada"*.

3.3 Principais Recursos

Assim como os demais programas de GD apresentados na seção 2.3, a versão atual do iGeom permite realizar todas as operações básicas de Geometria Dinâmica, como por exemplo: (a) criar objetos geométricos como pontos, retas, semi-retas, segmentos, circunferências, polígonos, áreas, medidas dinâmicas como ângulos e distâncias; (b) opções de edição: esconder/mostrar, remover ou desfazer remoção, criar textos, rastrear e modificar as características dos objetos; (c) opções de gravação/recuperação de arquivos em diferentes formatos; (d) e outros recursos "avançados" como isometrias, perpendiculares e paralelas.

Além das características usuais dos programas de GD, o iGeom possui algumas características que não são encontradas freqüentemente em outros programas de GD. Entre elas destacamos: a geração de "scripts" (ou "macros") recorrentes, a fácil exportação para Web, e desde o início de 2004, a abertura de múltiplas áreas de desenho, a autoria e validação automática de exercícios e a comunicação com servidores (visando seu emprego em sistemas gerenciadores de cursos Web).

Tabela 3.1: Recursos de alguns dos programas de GD

Programa	Portável	ADDs	Script	Rec	Web	AA	Com	Licença
iGeom	X	X	X	X	X	X	X	Gratuito
Cabri		X	X					Comercial
C.a.R.	X		X		X	X		GNU[1]
Cinderella	X		X		X	X		Comercial
GSP		X	X	X				Comercial
Tabulae	X	X			X		X	Comercial

Na tabela 3.1 destacamos alguns dos recursos existentes no iGeom em comparação aos programas de GD apresentados na seção 2.3. A coluna "Portável" refere-se aos programas que podem ser executados em qualquer plataforma (atualmente restringindo-se àqueles implementados em Java). A coluna "ADDs" refere-se àqueles que permitem a abertura de múltiplas áreas de desenho. Na coluna "Script" encontram-se aqueles que possuem recursos para a criação de *scripts* (macros). Na coluna "Rec" encontram-se aqueles que permitem a criação de *scripts* recorrentes. Na coluna "Web" estão aqueles que permitem o uso irrestrito de seus recursos diretamente em páginas Web. Na coluna "AA" estão aqueles que possuem recursos para autoria e

[1] *GNU General Public License* - permite livre distribuição e modificação do programa e seu código fonte. Para maiores informações consulte o endereço: http://www.gnu.org/copyleft/gpl.html.

validação automática de exercícios. Na coluna “Com” estão àqueles que possuem recursos de comunicação. E finalmente, a coluna “Licença” refere-se ao tipo de licença que cada um dos programas possui.

3.3.1 Scripts

No iGeom, como em qualquer outro programa de GD que possui este recurso, um *“script”* (macros) pode ser entendido como um roteiro (algoritmo) para realizar uma construção (como visto na seção 2.2). Uma grande vantagem na implementação dos *script* no iGeom (e no GSP) é a possibilidade de fazer uma chamada recorrente. A recorrência é caracterizada pela aplicação da própria função em sua definição. O exemplo muito conhecido de função recorrente é função fatorial f(n) = n!, que é definida por: (equação 3.1).

$$f(n) = \left\{ \begin{array}{ll} 1 & , se\ \ n = 0 \\ n * f(n-1) & , se\ \ n > 0 \end{array} \right\} \tag{3.1}$$

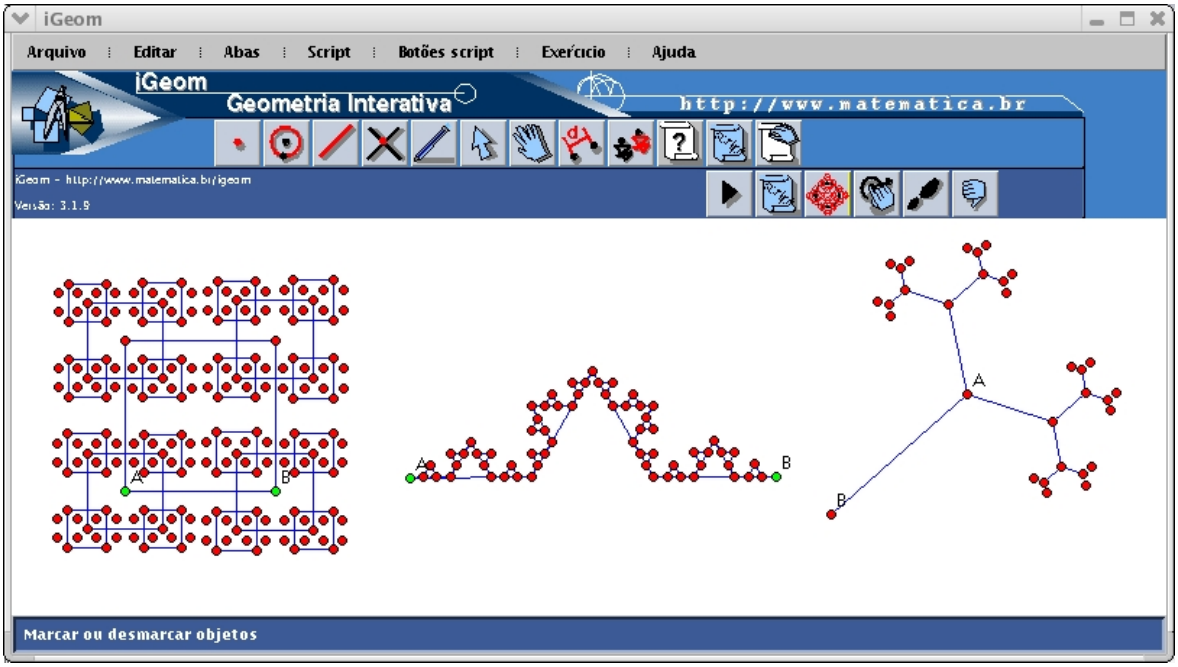

Figura 3.5: Fractais criados no iGeom

Os *scripts* recorrentes permitem vários usos didáticos. Por exemplo, o aluno poder construir um *script* que descreva um fractal (Figura 3.5) ou tentar descobrir propriedades usando uma

abordagem do tipo caixa preta (o professor disponibiliza o *script*, cabendo ao aluno analisar os resultados de sua aplicação).

Como um *script* pode ser entendido como um algoritmo, este pode ter ou não parâmetros. No iGeom, a ferramenta para a geração de *script* deduz os parâmetros automaticamente a medida que as construções são "anotadas". Outra característica deste recurso no iGeom é a possibilidade de disparar algum outro *script* já gravado para construir um novo.

3.3.2 Exportação para Web

Uma vez que o iGeom foi projetado para trabalhar também via Web (na forma de *applet*), é muito simples produzir construções geométricas que podem ser incorporadas em páginas na Internet. Assim, uma vez feita a construção é possível gravá-la utilizando o recurso *"Gerar applet"* disponível nas opções de menus, e o iGeom se encarrega de criar uma página Web padrão, contendo todo o código HTML[2] necessário para apresentar a construção na Web.

Para publicar a página HTML na Web, é necessário colocar esta página em algum diretório do servidor Web, acessível pela Internet. Neste mesmo diretório deve-se colocar também uma cópia da versão do iGeom que pode ser obtido em *http://www.matematica.br/igeom/iGeom.jar*.

```
<applet codebase="." archive="iGeom.jar" code="IGeomApplet.class"
        WIDTH=720 HEIGHT=590 alt="...">
 <param name="BGCOLOR" value="#eeffee">
 <param name="igeom" value="# igeom: http://www.matematica.br!
[iGeom versão 3.1.12]!
[ versao: 3.1.12 ]!
[[Mar 17, 2005 8:06:45 AM; seiji]]!
[0:1.1, 1:7, 2:0] - iGeom versão 3.1.12!
[Mar 17, 2005 8:06:45 AM; seiji]!
{1:0, 0:0, 2:314.0 -209.0, 3:2 3, 4:A 0.78539816339 0, 6:1, 5:-167119, 7:0}!
{1:1, 0:0, 2:393.0 -215.0, 3:2 7 3, 4:B 0.785398163 0, 6:1, 5:-167119, 7:0}!
{1:2, 0:3, 2:0 1, 3:4 5 7, 4:c0 0.785398163397448 0, 6:1, 5:-1677696, 7:0}!
{1:3, 0:3, 2:1 0, 3:4 5 7, 4:c1 0.785398163397448 0, 6:1, 5:-167796, 7:0}!
{1:4, 0:1, 2:2 3 2, 3:7 6, 4:C 0.785398163397448 0, 6:1, 5:-65536, 7:0}!
{1:5, 0:1, 2:2 3 1, 3:6, 4:D 0.785398163397448 0, 6:1, 5:-65536, 7:0}!
{1:6, 0:4, 2:4 5, 3:, 4:r 0.5 0, 6:1, 5:-16776961, 7:0}!
"></applet>
```

Figura 3.6: Parte do código gerado ao exportar a construção da mediatriz

A página gerada pelo iGeom, ao ser aberta por um navegador Web, apresentará o formato padrão das páginas do iMática (*http://www.matematica.br*). Entretanto, não é difícil para

[2]HTML - HyperText Markup Language, linguagem padrão para publicação de "híper-documentos" na Web.

o usuário (com um pouco de experiência) transportar sua construção para uma outra página HTML qualquer. Para isso, basta transferir um pedaço de texto HTML gerado pelo iGeom da seguinte forma: (a) abrir o arquivo gerado pelo iGeom em um editor de texto (como o *Emacs* no Linux/Unix ou o *Notepad* no Windows) e também o outro arquivo HTML (no qual deseja inserir sua construção); (b) identificar os comandos relativos ao iGeom, como apresentado na Figura 3.6; (c) copiar este código, que está entre `<applet ... </applet>` (incluindo estas marcas), para o arquivo de destino; (d) Gravar o arquivo modificado.

3.3.3 Recursos Desenvolvidos

Nesta seção apresentaremos alguns dos principais recursos que desenvolvemos no iGeom durante a realização deste trabalho de mestrado.

Múltiplas Áreas de Desenho

A grande vantagem em dispor de múltiplas áreas de desenho (abas) é a possibilidade de "interligação" entre as mesmas. Por exemplo, é possível copiar partes de uma construção realizada em uma área de desenho para outra. Outra vantagem é facilitar a comparação entre diferentes construções. Antes da implementação das múltiplas abas era necessário abrir diferentes instâncias do iGeom, uma para cada construção.

Um outro motivo que nós levou a implementação deste recurso foi facilitar o desenvolvimento de recursos para cooperação, que permitirá que as construções sejam compartilhadas de alguma forma por mais de um usuário (conectados através de uma rede de Internet ou Intranet).

Uma característica importante na ferramenta de copiar/colar implementada no iGeom é a manutenção das propriedades geométricas. Dois exemplos podem ilustrar bem como implementamos esta opção: se um objeto do tipo ponto estiver sobre uma reta, ao copiar este ponto, a reta também será copiada, e contudo, ela não ficará visível na área de desenho que recebeu a cópia (este objeto estará lá como um objeto escondido); se um ponto de interseção for copiado, os objetos que definem a interseção também serão copiados e ficarão escondidos.

O acesso às várias áreas de desenho no iGeom é realizado por meio das **abas**, conforme mostra a Figura 3.7. Cada área de desenho possui sua aba correspondente. A aba que está em uso recebe uma coloração diferente e fica visivelmente num plano "acima" das outras demais abas.

Figura 3.7: Múltiplas abas no iGeom

Autoria e Validação Automática

Baseado na dinâmica dos programas de GD e na estrutura interna do iGeom, desenvolvemos dois recursos interligados de particular utilidade ao professor. Um para criar exercícios e outro para validá-los automaticamente. No capítulo 5 mostramos a técnica utilizada para fazer a validação automática e detalhes de implementação destes dois recursos.

Nossa implementação permite criar exercícios que podem ser executados pela Web ou no próprio aplicativo do iGeom, diferentemente do Cinderella e do C.a.R. que permitem apenas que o exercício seja executado e validados em páginas na Web. Um dos motivos desta limitação nos programas citados decorre da dificuldade em liberar/bloquear as suas ferramentas (botões). Entretanto, como o projeto inicial do iGeom já previa tal possibilidade, foi razoalmente simples implementarmos a liberação e bloqueio de botões no iGeom, seja na forma de aplicativo quanto em *applet*.

Utilizando o recurso de validação automática de exercícios, para cada solução considerada incorreta, encontramos uma posição da construção na qual fica claro que a solução possui algum erro. Esta configuração (instância) será chamada aqui de **contra-exemplo**. O uso de contra-exemplos auxilia tanto o professor para verificar o problema na construção do aluno, quanto ao próprio aluno que tem a possibilidade de visualizar seu erro.

A autoria e a validação automática, em conjunto com o recurso de exportação para Web, viabilizaram a criação de exercícios interativos que podem ser utilizados em páginas Web de livre acesso. Através deste conjunto de ferramentas o professor pode produzir um conjunto de páginas Web com diversos exercícios e oferecer aos seus alunos uma validação imediata de suas construções sem a necessidade de verificar cada construção pessoalmente.

Um exemplo prático da utilização destes recursos pode ser encontrado no iMática a partir do endereço *http://www.matematica.br/igeom/docs/exemplo1/*. Este é um *site* criado por Sandra Cairolli como trabalho final da disciplina MAC118 - *Noções de ensino de matemática usando o computador*, em 2004. Este trabalho possui diversas atividades (divididas em aulas, tópicos e exercícios) para ensino de Geometria (Figura 3.8). Todas elas podem ser realizadas diretamente

na Web e o resultado da validação (se está correta ou não) de cada solução é fornecida pelo iGeom após o aluno marcar sua resposta e clicar no botão de “envio de resposta”.

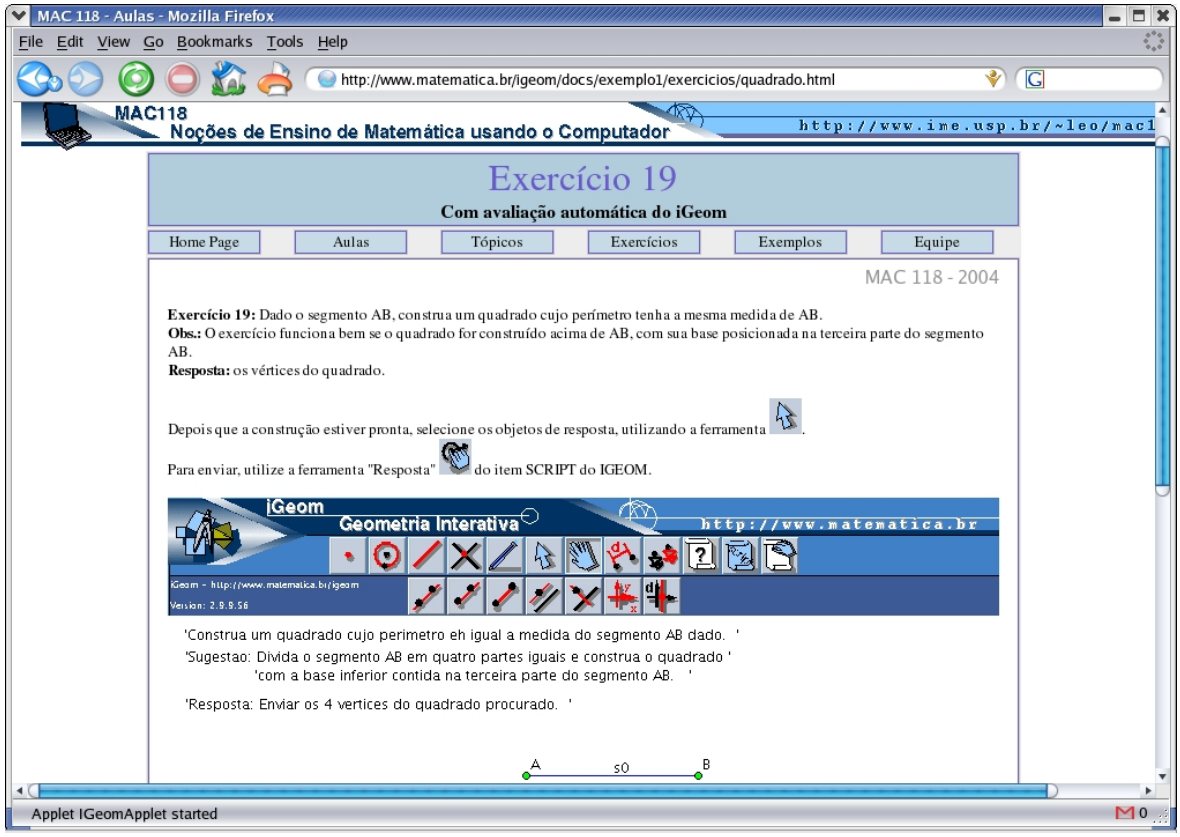

Figura 3.8: Exemplo de atividade para uso via Web desenvolvida com o iGeom

Comunicação

Os recursos de autoria e validação automática podem ser ainda mais interessantes se estiverem conectados com algum sistema de gerenciamento de curso. Para fazer este “casamento” dotamos o iGeom com o recurso de **comunicação** que permite ao programa enviar e receber mensagens pela Internet através de uma conexão padrão.

A grande aplicação deste recurso é incorporar o iGeom a um sistema gerenciador de cursos pela Web. Assim, pode-se disponibilizar exercícios para o aluno a medida que ele vai resolvendo outros exercícios. Além disso, o professor também pode usar o gerenciador com o iGeom para publicar novos exercícios e examinar as soluções (corretas e incorretas) enviadas pelos alunos.

Os dois programas de GD que sabemos dispor de autoria e validação automática, o Cin-

derella e o C.a.R., apresentam algumas restrições no uso destas ferramentas: só é possível criar exercícios utilizando o programa no formato aplicativo e só é possível resolvê-los via *applets*. No iGeom estes recursos são de livre acesso tanto pela Web (via applet) quanto fora dela (via aplicativo).

Através destes recursos, um professor pode criar cursos de geometria, presenciais ou mesmo à distância, com exercícios interativos para seus alunos. Quando um aluno terminar um exercício o iGeom envia uma mensagem ao servidor contendo algumas informações sobre o desempenho do aluno durante a realização do exercício. Por exemplo, atualmente são enviados informações ao servidor com o resultado da validação (feita pelo iGeom) sobre a solução do aluno, um contra-exemplo quando o exercício é avaliado como incorreto e quais foram os botões mais utilizados. Todas estas informações podem ser armazenadas pelo servidor para posterior análise.

O mecanismo de comunicação do iGeom e seu uso em cursos na Web serão discutidos mais detalhadamente no capítulo 4.

Capítulo 4

Educação a Distância e a Geometria Dinâmica

> "*What I really want to point out is that we need software that makes it easy for everybody to create real, interactive content for the Internet.*"
>
> (Kortenkamp, 1999)

Atualmente, segundo Hentea et al. (2003) e Litto et al. (2004), é grande a popularização da **educação a distância** (EAD) via Internet e o seu consequente impacto nos métodos de ensino, que precisam ser adaptados ou criados para se adequar a esta modalidade de ensino. Cada vez mais escolas de todos os continentes utilizam esta prática de ensino para melhorar ou cobrir as deficiências do ensino tradicional. Algumas das vantagens mais citadas são: a possibilidade de alcançar aqueles que moram em locais afastados dos centros urbanos e permitir o ensino individualizado utilizando o ritmo de aprendizagem de cada aluno.

O uso da EAD não é recente. Segundo o trabalho de Nunes (1994), existe uma longa história sobre esta prática de ensino cuja data de inicio antecede o século XVIII. Hoje em dia o termo "educação a distância", esta vinculado ao processo de ensino-aprendizagem mediado por algum tipo de tecnologia (televisão, computador, Internet, entre outras). De acordo com o *Instructional Technology Council*[1] (ITC), educação a distância é:

[1]O ITC é um conselho afiliando à Associação Americana de Faculdades que representa as instituições de ensino superior nos Estados Unidos e no Canada. Mais informações sobre o ITC estão disponíveis no endereço http://www.itcnetwork.org/definition.htm

> *"O processo de estender o aprendizado, ou a oportunidade de compartilhar recursos educacionais, para regiões distantes de uma sala de aula, edifício ou localidade, para uso em outra sala de aula, edifício ou localidade, através de recursos como vídeo, áudio, computador, comunicações multimídias, ou a combinação destes com outros métodos tradicionais de distribuição."*

Neste trabalho, estamos apenas nos referindo a EAD realizada pela internet, ou seja, como sinônimo de *e-learning*.

A EAD teve grande impacto mundial com o aparecimento da Internet e da World Wide Web. A World Wide Web, ou apenas Web ou WWW, é um sistema de armazenamento, recuperação e troca de informação pela Internet, originado no *European Center for Nuclear Research* (CERN), no início dos anos 90. O Nascimento da WWW surgiu com a proposta de um protocolo de comunicação com recursos gráficos e interligações (*links*). Para gerenciar estes recursos o CERN definiu uma linguagem de marcação (***tags***) para publicação de conteúdo na Web, a *HyperText Markup Language*, também conhecida como **HTML**, e em 1993, o *National Center for Supercomputing Applications* (NCSA) apresentou o primeiro navegador Web da história, o Mosaic[2]. Do mesmo modo que o sistema gráfico com o mouse criado no laboratório da Xerox revolucionou e impulsionou o uso dos computadores, o Mosaic revolucionou e impulsionou o uso da Internet.

Do ponto de vista didático a Web possibilita que professores e alunos compartilhem um novo espaço de trabalho, que podemos chamar de "virtual". Neste espaço virtual existem formas de comunicação que ocorrem de modo síncrono (através de chats, mensagens instantâneas, etc) e outras de modo assíncrono (através de fóruns de discussão, emails, etc).

Todos estes recursos são genéricos, no sentido de serem utilizáveis em quaisquer cursos oferecidos na Web. Os gerenciadores de cursos pela Web, como o WebCT[3], Moodle[4], Teleduc[5], AulaNet[6], e outros, possuem estes recursos. Entretanto estes não dispõem de recursos especializados para cursos específicos (como de Geometria), nem têm a possibilidade de incorporar facilmente tais recursos.

Dentre os principais benefícios da EAD apresentados por, Lawhead et al. (1997) e Hentea et al. (2003), destacamos os seguintes:

[2]O Mosaic está disponível em *http://archive.ncsa.uiuc.edu/SDG/Software/Mosaic/*
[3]WebCT - Web Course Tools. Disponível em *http://www.webct.com/*
[4]Moodle - Course Management System. Disponível em *http://www.moodle.com/*
[5]Teleduc - Ambiente de suporte para ensino-aprendizagem a distância. *http://www.teleduc.com.br*
[6]Sistema AulaNet de Ensino a Distância. Disponível em *http://www.aulanet.com.br*

- Possibilitar que cursos atinjam comunidades em locais remotos;
- Permite o aprendizado “individual”, onde cada aluno estuda em seu próprio ritmo, no local e tempo que lhe seja conveniente;
- Facilidade para personalizar o aprendizado como, por exemplo, montar diferentes grandes curriculares;
- Maior facilidade de distribuição do material e sua posterior reprodução.

Nos últimos anos diversas universidades e empresas do Brasil e do exterior têm criado cursos para EAD (Litto et al., 2004). Podemos notar a proliferação cursos técnicos e de extensão, e do aparecimento de alguns de graduação e até de mesmo de pós-graduação oferecidos à distância. Universidades de grande prestígio como a Universidade de Harvard[7] e o Instituto de Tecnologia de Massachusetts[8] (MIT), nos Estados Unidos, oferecem diversos cursos à distância. Como observa Litto (Litto, 2003): *“As mais importantes universidades do mundo fazem amplo uso da educação a distância”*.

No Brasil, segundo o estudo realizado por Litto et al. (2004), o número de trabalhos relacionados à EAD aumentou de 62, em 1999, para 208, em 2003, sendo que em 2002 foram mais de 280 trabalhos apresentados nesta área (veja a tabela 4.1). Segundo Litto et al. (2004), a redução de produtividade de 2002 para 2003 pode ser o resultado da “explosão da bolha digital”, fenômeno que ocorreu em 2001 quando a bolsa de valores de tecnologia, a NASDAQ, entrou em colapso.

Tabela 4.1: Quantidade de publicações brasileiras relacionadas a EAD entre 1999-2003 (Litto et al., 2004)

Ano de Publicação	Dissertação/Tese	Artigos	Total
1999	31	31	62
2000	54	75	129
2001	147	13	160
2002	158	130	288
2003	101	107	208

Como bem observa Hentea et al. (2003), a educação a distância não veio substituir os cursos presenciais, mas sim, preencher as lacunas e falhas que esta forma tradicional apresenta.

[7] O programa de EAD de Harvard está disponível em *http://www.extension.harvard.edu/DistanceEd/*
[8] O programa de EAD do MIT está disponível em *http://lfmsdm.mit.edu/sdm/distance.html*

Apesar dos grandes benefícios que a educação a distância pode oferecer, o seu uso inadequado pode provocar resultados negativos. Do ponto de vista do professor, pode gerar dificuldade para preparar e ministrar as aulas virtuais e, do ponto de vista do aluno, pode desmotivá-lo e criar frustrações em relação ao curso, aumentando o índice de desistência. Segundo Kaczmarczyk (2001) e Berge et al. (2002), as principais preocupações relacionadas à expansão da EAD são: a queda na qualidade de ensino, a falta de preparo pedagógico por parte dos professores e a falta de disciplina/motivação dos alunos.

Segundo Hentea et al. (2003), devido a esta preocupação, nos ambientes de EAD, grande parte do esforço exercido pelos pesquisadores está direcionada para : (a) o desenvolvimento e estudo de novos métodos de ensino adequados a esta nova visão educacional; (b) a criação de conteúdo interativo de boa qualidade; (c) aumentar a motivação do aluno; (d) prover novas estratégias de avaliação do aluno; (e) fazer uso efetivo das tecnologias.

4.1 Interatividade e o Impacto na EAD

Criar conteúdo para EAD não é uma tarefa simples, pois além de demandar o conhecimento da área também é necessário o conhecimento das ferramentas disponíveis para produção e gerenciamos do curso. Entretanto, ainda existe uma dificuldade adicional: o método de ensino. Transformar livros didáticos em textos que podem ser acessados pela Internet, não garante o sucesso dos cursos à distância. Isso porque tanto o aluno, quanto o professor enfrentam barreiras no processo de ensino-aprendizagem à distância devido a diversos fatores. Dentre eles destacamos (Litto, 2003; Zirkle, 2004):

- Falta de motivação pessoal;
- Avaliação demorada ou inadequada;
- Falta de contato com o professor;
- Despreparo técnico do aluno ou do professor;
- Sensação de alienação e isolamento;
- Conteúdo desorganizado e em formato inadequado;
- Falta de suporte técnico.

Segundo Hentea et al. (2003), muitos alunos que participaram de cursos à distância possuem reclamações sobre os métodos de ensino e avaliação, além do formato e organização do

conteúdo oferecido. Os trabalhos de Hentea et al. (2003) e Hijazi (2003) apontam que um dos principais fatores do insucesso dos cursos à distância é a falta de interação entre aluno-aluno, aluno-professor e também entre aluno-conteúdo. Estes trabalhos também relatam que o conteúdo com bom nível de interação produzem maior compreensão.

Cavanaugh (2001) em seu estudo realizado com mais de 900 alunos do ensino fundamental e médio indica que em cursos à distância a combinação do uso de programas interativos e o oferecimento de aulas presenciais esporádicas podem aumentar consideravelmente o aprendizado.

De acordo com Hijazi (2003), os benefícios da inserção de programas interativos em ambientes de educação a distância são:

- Possibilidade de usar diferentes formas de apresentação do mesmo material;
- Possibilita a análise contínua do curso;
- Pode aumenta as taxas de aprendizado;
- Permite criar ambientes para aprendizagem no modo autodidata;
- Provê recursos para inibir a sensação de isolamento ou alienação;
- Permite o armazenamento das interações do aluno com o conteúdo.

Atualmente as principais ferramentas de gerenciamento de conteúdo para ambientes de educação a distância possuem algumas ferramentas interativas para interação aluno-aluno ou aluno-professor. As mais comuns segundo Guerra (2000) são: os emails, chats, fóruns e listas de discussão.

No entanto, ainda são raras as ferramentas para EAD que oferecem a interação entre aluno-conteúdo e que não sejam baseadas em texto (por exemplo, lousas virtuais). Este problema é crítico quando pensamos no ensino de Matemática De acordo com Santos & Sola (2001), para superar este problema é necessário desenvolver recursos específicos para cada área do conhecimento considerando suas diferentes formas de apresentar o conteúdo.

4.2 A Geometria Dinâmica na Internet

Segundo Jones (2003), o ensino de Matemática pela Internet ainda está dando seus primeiros "passos" quando comparada às outras áreas de pesquisa em educação Matemática. Contudo,

este mesmo autor concorda que com o aparecimento das ferramentas interativas, que podem ser manipuladas diretamente pela Internet, este cenário esta começando a se modificar.

O *applet* é atualmente um recurso muito utilizado para prover interatividade em páginas na Web. No ensino de Matemática pela Internet este recurso abriu novos caminhos proporcionando o desenvolvimento de páginas interativas com gráficos, animações, construções geométricas e a possibilidade de interação em tempo real.

Para usufruir o benefício dos *applets*, os programas de GD mais populares como o Cabri e o GSP, que não possuíam suporte para uso via Web, criaram mecanismos para transformar o conteúdo desenvolvido nestes programas em *applets* interativos. Contudo, no caso do Cabri e do GSP, estes *applets* não permitem utilizar todos os recursos da GD (por exemplo, não é possível criar objetos, editá-los ou escondê-los).

Com o aparecimento de programas desenvolvidos em Java como o Cinderella, C.a.R. e o iGeom, esta limitação foi superada, pois os *applets* criados por estes programas permitem o uso praticamente integral de seus recursos via Web.

4.2.1 Criando Páginas Interativas

Na seção 3.3.2 foi apresentado uma funcionalidade do iGeom que permite exportar para Web as construções geométricas criadas neste programa utilizando uma opção disponível nas opções de menu. Outros programas escritos em Java[9] como o Cinderella e o C.a.R., possuem recurso semelhante. Além destes, tanto o Cabri quanto o GSP em suas versões mais recentes disponibilizam este recurso em seu programa principal[10].

Da mesma forma que o iGeom, uma página Web criada pelos programas citados acima deve ser acompanhada de um *applet* para que funcione corretamente. No Cinderella, ao criar uma página Web, um *applet* (cindyrun.jar) é inserido automaticamente no mesmo diretório onde esta foi criada. Os programas como o Cabri e o GSP, que inicialmente não foram concebidos para serem utilizados pela Web, criaram *applets*, o CabriJava e o JSP Applet, para interpretar suas construções. Enquanto no *applet* iGeom é possível utilizar praticamente todas as suas ferramentas, tanto CabriJava quanto o JSP Applet, permitem apenas a movimentação dos objetos.

[9] O Tabulae, segundo seus autores, não possui esta funcionalidade disponível em sua versão de distribuição padrão.

[10] Em versões mais antigas do Cabri e do GSP, para criar uma página Web era necessário utilizar outros programas, o CabriWeb e o JavaSketchpad Converter (JSP), respectivamente.

Para Jiang (1999), a possibilidade de explorar os conceitos da GD pela Internet, abriu novas possibilidades de aprendizagem permitindo que o aluno explore os conceitos dados em aula em sua própria casa, eventualmente eliminando dúvidas e realizando novas investigações. Muito além de simples páginas interativas, a GD na Web oferece ambientes virtuais propícios para o incentivo da prática construtiva e provê meios para o compartilhamento e o armazenamento de conteúdo Web.

Apesar de encontrarmos vários *sites* com propósitos educacionais que utilizam os *applets* criadas por programas de GD, como por exemplo o *The Math Forum*[11] e o *MathsNet*[12], são poucos os ambientes de EAD que fazem uso deste recurso. Isso ocorre devido a falta de recursos em programas de GD para: publicação de conteúdo para Web, autoria e validação automática de exercícios e comunicação.

Com estes recursos podemos:

1. **Publicação de Conteúdo**: Gerar de modo automático páginas que podem publicadas na Web;

2. **Autoria e validação automática de exercícios**: Com estes recursos, diminuímos a carga de trabalho do professor, tanto para preparar conteúdo quanto para avaliar as soluções dos alunos;

3. **comunicação**: com este recurso as interações entre o aluno e o *applet* podem ser armazenadas facilitando uma análise mais detalhada pelo professor.

4.3 O iGeom e a EAD

Atualmente existem vários sistemas complexos para gerenciar cursos pela Web, inclusive alguns gratuitos, como o Teleduc e o Moodle. Entretanto estes ambientes são desprovidos de recursos especializados para o aprendizado de conteúdos específicos, como a Geometria.

Como já citado, a possibilidade de integrar o iGeom nestes sistemas oferece grandes vantagens para o aluno e para o professor. Ao aluno, além da possibilidade de realizar os exercícios diretamente pela Web, podemos apresentar-lhe o resultado da validação de sua solução quase que instantaneamente. Ao professor, oferecemos recursos que facilitam e agilizam o processo

[11] http://mathforum.org/dynamic/
[12] http://www.mathsnet.net/dynamic/

de criação e validação de conteúdo e, além do envio de dados relacionados à interação do aluno com o programa para uma posterior análise.

4.3.1 A Comunicação do iGeom

A comunicação entre o iGeom e o servidor é feita através de parâmetros inseridos pelo servidor na página Web e da troca de mensagens entre o *applet* e o servidor via conexão **HTTP** utilizando o método **POST**[13]. A escolha dos protocolos HTTP e POST é devido a sua popularidade e por estar disponível em qualquer plataforma.

A comunicação do iGeom, em conjunto com os recursos anteriormente apresentados na seção 3.3, traz diversas vantagens. Dentre elas destacamos:

- um professor pode produzir os exercícios em sua máquina e enviá-los ao servidor ou criá-los diretamente no *applet* acoplada ao sistema gerenciador (Figura 4.1);
- enviar os dados relacionados a resolução de um exercício realizado. Por exemplo, após um aluno resolver o exercício e o iGeom validá-lo, é possível enviar este resultado ao servidor acompanhado de um contra-exemplo, caso a solução seja considerada incorreta (Figura 4.1).
- receber dados do servidor e, desta forma, poder direcionar a próxima página Web que será exibida após a realização de um exercício. Por exemplo, quando um aluno completa um exercício, dependendo do resultado da validação, ele pode ser encaminhado para uma página Web diferente. Este recurso permite ainda mostrar ou bloquear os botões do iGeom e, de modo imediato, oferecer ou não o resultado da validação da solução para o aluno.

Para que o envio de mensagens ocorra, precisamos indicar o endereço do servidor que irá receber as mensagens do iGeom. Essa indicação é feita na forma de parâmetro[14] descrito na página HTML da seguinte forma:

```
<param name="enderecoPOST" value="http://aqui_vai_o_endereço">
```

Com esta informação, é possível realizar uma conexão HTTP entre o *applet* e este endereço no servidor. O envio de mensagens é feito utilizando a codificação POST que permite o envio de variáveis na forma de cadeias de caracteres. Dessa forma, podemos enviar diferentes

[13] Informações detalhadas sobre o método POST podem ser encontradas em http://www.w3.org/MarkUp/
[14] O parâmetro nada mais é que uma *tag* HTML que fornece algum dado para a *applet*.

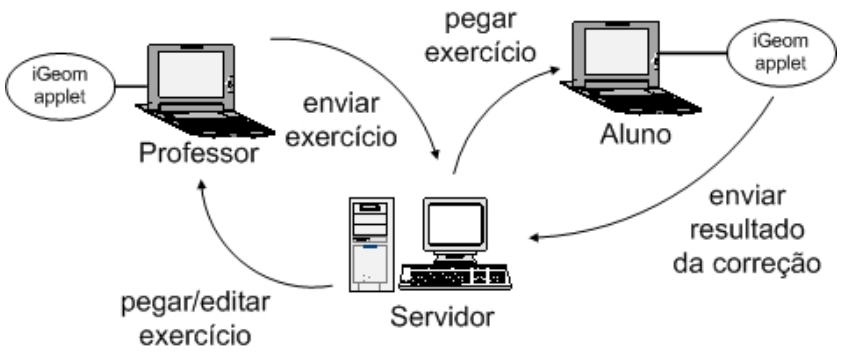

Figura 4.1: Esquema de comunicação entre um servidor e o iGeom em sistemas Web

informações para o servidor em uma mesma mensagem. Observe que neste caso o programa localizado no servidor deve ser o responsável pelo tratamento adequado dos dados recebidos, fazendo a decodificação necessária para recuperar os valores das variáveis enviadas.

Um exemplo de envio de mensagens ocorre após a validação de um exercício realizado em uma página Web. Quando o botão de enviar exercícios é acionado, o iGeom faz uma requisição para se conectar um endereço no servidor. Ao receber a autorização é possível fazer o envio de dados. Estes dados são armazenados pelo iGeom em diferentes variáveis. Cada variável é definida por um conjunto de caracteres que serão posteriormente "empacotadas" utilizando o método POST. Este método faz a codificação destes caracteres, para que seja possível o envio de dados através de uma conexão HTTP.

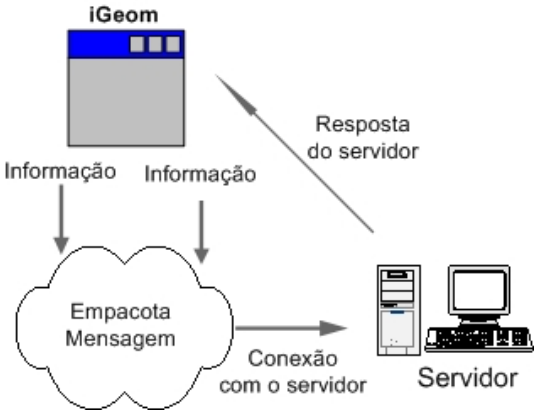

Figura 4.2: Envio de mensagens entre o iGeom e o servidor

Quando o envio de dados ocorre, o programa servidor irá armazenar as mensagens recebidas em variáveis locais para que sejam manipuladas pelo programa gerenciador. Deste modo, através desta mesma conexão HTTP, o servidor pode requisitar a troca da página Web ativa para uma nova página, dependendo dos dados da mensagem recebida (Figura 4.2) e, posteriormente, anotar estes dados em um banco de dados (por exemplo, o MySql).

Atualmente as variáveis que o iGeom manipula são as seguintes:

$envWebValor: Esta variável indica o resultado da validação do exercício. Se seu valor for igual a 0 (zero) então o exercício foi avaliado como incorreto, e se for igual a 1 (um) o

exercício foi avaliado como correto.

$envWebArq: Esta variável possui dupla funcionalidade, uma quando o professor esta construindo um exercício e outra quando um aluno está resolvendo um exercício. No caso do professor, ao criar um exercício diretamente em ambiente Web, esta variável armazena o arquivo contendo todas as informações do exercício (inclusive o gabarito) que será utilizado posteriormente pelo aluno. No caso do aluno, esta variável armazena o arquivo contendo toda a construção realizada por ele durante a resolução de um exercício. Se a solução é considerada incorreta, então será armazenado uma configuração na qual fique visível o problema, senão a solução é armazenada na configuração na qual o aluno resolveu o exercício.

$envWebGeoResp: Esta variável é utilizada para indicar quais foram os objetos da solução de um exercício selecionados como resposta.

$envWebGeoOuvidor: Nesta variável são inseridos dados da interação do usuário com a interface do iGeom. Por exemplo, quais os botões mais utilizados e a forma de interação utilizada (se ação+seleção ou seleção+ação).

$envWebArquivoCurso: Ao criar um exercício diretamente em ambiente Web, da mesma forma que a variável $envWebArq, esta variável armazena o arquivo contendo todas as informações do exercício que será utilizado posteriormente pelo aluno. Contudo está variável recebe uma codificação especial na qual apenas o iGeom na forma de *applet* consegue utilizá-lo.

Por questões de segurança, as variáveis *$envWebArq* e *$envWebArquivoCurso*, quando carregam respectivamente os dados da solução de um exercício realizado pelo aluno e o gabarito de um exercício construído pelo professor, recebem uma codificação mais “sofisticada”, que só permite sua abertura pelo iGeom na forma de *applet*.

A razão disso é devido a dois fatores: o formato dos arquivos gerados pelo iGeom para serem abertos pelas versões aplicativo ou pelo applet, são similares; e a versão aplicativo do iGeom permite que o professor tenha acesso ao seu gabarito, para eventuais edições.

Deste modo, ao publicar um exercício para o aluno é necessário bloquear sua abertura pelo iGeom aplicativo, pois em caso contrário, o aluno conseguiria acesso ao gabarito do professor (e portanto à resposta) tranferindo a descrição do exercício para o aplicativo.

Na tabela 4.2 apresentamos parte do código iGeom e sua codificação.

Tabela 4.2: Passos para codificação de algumas variáveis no iGeom

Código iGeom

```
{1:0, 0:0, 2:321.0 -212.0, 3:2 3, 4:A 0.7853981633974483 0, 6:1, 5:-16711936, 7:0}
{1:1, 0:0, 2:392.0 -219.0, 3:2 7 3, 4:B 0.7853981633974483 0, 6:1, 5:-16711936, 7:0}
{1:2, 0:3, 2:0 1, 3:4 5 7, 4:c0 0.7853981633974483 1, 6:1, 5:-16776961, 7:1}
```

Cadeia de Caracteres Codificada

```
7b313a302c20303a302c20323a3332312e30202d3231322e302c20333a3220332
c20343a4120302e373835333938313633333937343438332030 2c20363a312c20
353a2d31363731313933362c20373a307d0a7b313a312c20303a302c20323a333
9322e30202d3231392e302c20333a32203720332c20343a4220302e3738353339
383136333339373434383320302c20363a312c20353a2d31363731313933362c2
0373a307d0a7b313a322c20303a332c20323a3020312c20333a34203520372c20
343a633020302e373835333938313633333937343438332031 2c20363a312c20
353a2d31363737363936312c20373a317d0a7b313a332c20303a332c20323a312
0302c20333a34203520372c20343a633120302e373835333938313633333937734
34383320312c20363a312c20353a2d31363737363936312c2020302e373833a3
```

Na seção seguinte, apresentamos um sistema gerenciador de cursos pela Web que aplica os recursos apresentados nesta seção.

4.3.2 O SAW+iGeom

Com a crescente expansão da Internet e do uso de computadores no ensino, gerou-se uma grande demanda por cursos via Web, e conseqüentemente, por ferramentas que facilitem sua produção e reutilização. Além disso, características como a flexibilidade e a interatividade do sistema são essenciais para viabilizar aulas mais dinâmicas e interessantes.

A partir destas idéias, no final de 2003 foi iniciado o desenvolvimento, coordenado pelo professor Brandão no IME-USP, do **SAW - Sistema de Aprendizagem pela Web** (Brandão et al., 2004b). Este sistema utiliza uma arquitetura cliente/servidor, sendo que na parte cliente faz uso de aplicativos Java (*applets*). Este sistema está sendo desenvolvido em *PHP*, utilizando o gerenciador de banco de dados *MySQL*. O SAW faz parte do trabalho da aluna de mestrado Janine Gomes Moura no IME-USP (Moura & Brandão, 2004).

No início do primeiro semestre de 2004, o SAW com o iGeom foi utilizado por estudantes e professores em uma disciplina obrigatória oferecida para o curso de licenciatura em matemática do IME-USP, *MAC118 - Noções de Ensino de Matemática Usando Computador*, ministrada em três turmas, uma diurna e duas noturnas, com 2 professores, 3 monitores e mais de 150 alunos.

Na Figura 4.3 está a atual interface do SAW+iGeom.

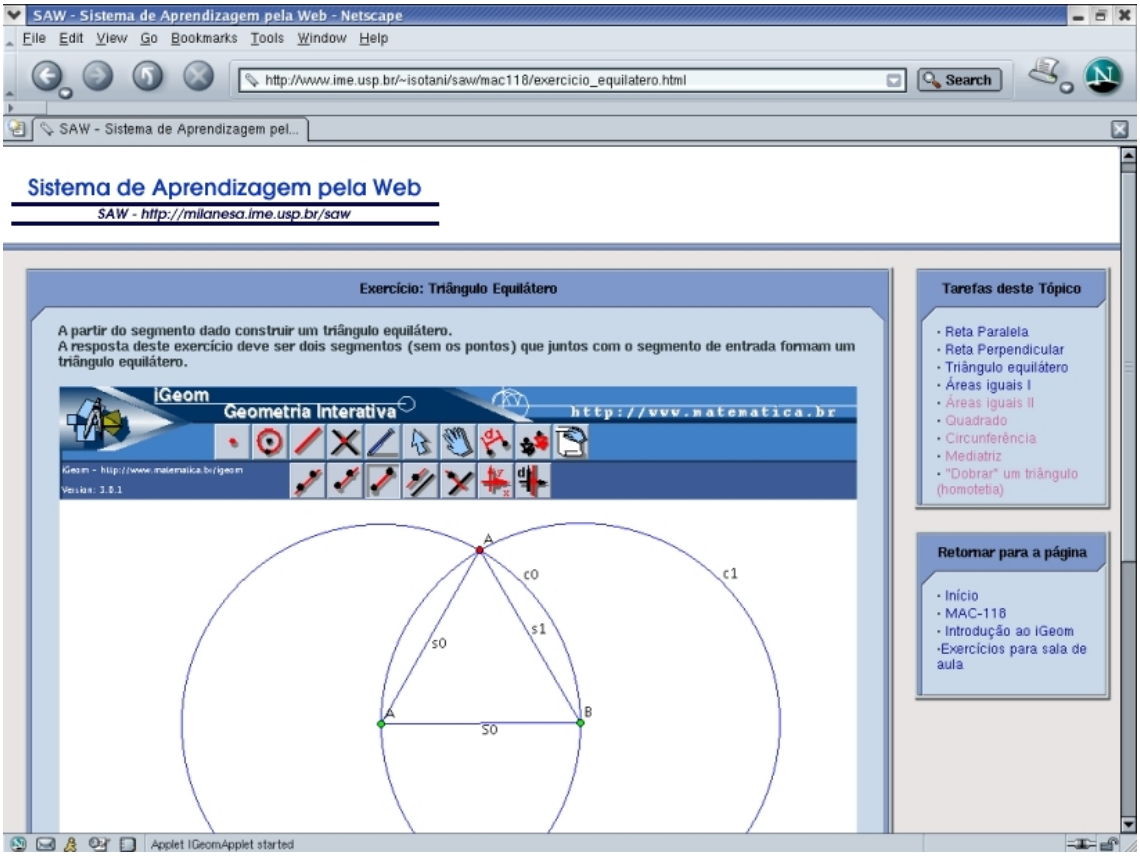

Figura 4.3: Interface do SAW+iGeom

Em edições anteriores da disciplina MAC118, todos os exercícios eram realizados utilizando o programa iGeom, mas sua correção era feita manualmente pelos monitores e professores. Devido ao número de alunos, a correção consumia grande parte do tempo dos monitores e o resultado da correção do exercício era entregue ao aluno duas ou três semanas após a realização do mesmo. Eram aplicados cerca **20** exercícios por semestre. Com o uso do SAW e da ferramenta de validação automática do iGeom, além de reduzir o trabalho de professores e monitores, foi possível aplicar mais de **40** exercícios, com a apresentação imediata do resultado da validação, além de permitir que os exercícios fossem realizados via Internet.

Através da aplicação da validação automática os alunos tiveram a possibilidade de tirar as dúvidas sobre a resolução do exercício imediatamente. Como afirma um aluno: "... *caso a construção estivesse certa, já estava enviada e caso estivesse errada, começaria novamente e tiraria as dúvidas na mesma hora...*".

Com o recurso de comunicação apresentado na seção 4.3.1, cada exercício realizado pelo aluno no sistema SAW+iGeom é armazenado em banco de dados para que o professor possa verificar posteriormente a construção do aluno. Atualmente, quando o exercício está incorreto, o iGeom envia a solução do aluno em uma configuração que facilita a visualização do erro (contra-exemplo). Caso o exercício esteja correto, é enviada a construção na configuração inicial.

Hoje o professor pode, no SAW+iGeom, criar os exercícios diretamente pela Web e os armazene em um banco de dados. Deste modo, facilitamos a reutilização dos exercícios criados, por diferentes professores e para diferentes cursos. Assim, é possível construir uma biblioteca de soluções distintas para um mesmo problema, além de uma com os erros mais freqüentes. Algumas das idéias envolvidas no desenvolvimento de uma biblioteca *On-line* de exercícios de Geometria foram discutidas por Barros & Santos (2000), Santos & Sola (2001), Santos et al. (2001) e Valente (2003).

No trabalho de Valente (2003), foi utilizada a versão **2** do iGeom, que não dispunha de autoria e validação automática, e por isso foi implementada uma solução intermediária com o gerenciador analisando as construções.

Capítulo 5

Autoria e Validação Automática de Exercícios

> *"O problema pode ser modesto, mas se ele desafiar a curiosidade e puser em jogo as faculdades inventivas, quem o resolver por seus próprios meios experimentará a tensão e gozará o triunfo da descoberta."*
>
> (Pólya, 1978)

O processo de aprendizado pode ser mais efetivo quando o aluno procura resolver, ele próprio, problemas adequadamente propostos. Como bem observa George Pólya (Pólya, 1978): *"A Matemática é a arte de resolver problemas ... e para resolver problemas é preciso resolver problemas"*. Sob este ponto de vista, é interessante oferecer ao aluno um bom número de problemas. Mas também é importante que o aluno receba rapidamente o retorno sobre sua solução. Entretanto, se o professor não dispuser de recursos auxiliares, precisará dispor de muito tempo para atender a estas demandas.

Estas demandas, que valem tanto para o ensino presencial quanto à distância, podem ser atendidas por sistemas computacionais que permitam a autoria e a validação automática de exercícios[1]. Uma das formas mais antigas de suporte à autoria e validação são os sistemas com questões do tipo múltipla-escolha, verdadeiro ou falso, ou de preenchimento de lacunas. Todos estes admitem uma validação rápida e simples, mediante a existência de um gabarito. Devido a esta simplicidade, muitos cursos à distância utilizam estes recursos para analisar o aprendizado de seus alunos (Gibson et al., 1995; Scapin, 1997).

A importância da rapidez na apresentação da validação da solução do aluno é destacada, por exemplo, nos trabalhos de Hara & Kling (1999), Kirby (1999) e Hentea et al. (2003). Estes

[1]Entende-se por exercício qualquer problema a ser solucionado.

trabalhos apontam que uma das principais frustrações dos alunos nos cursos à distância é a limitação ou a falta de uma validação/avaliação imediata.

5.1 Métodos de Validação Automática de Exercícios

O ato de validar é um processo complicado, exigindo testes que comprovam: a validade, a correção e a concordância com padrões previamente estabelecidos. Do mesmo modo, a validação automática não é trivial, podendo exigir algumas heurísticas ou simplificações desta tarefa.

A dificuldade sobre a qual nos concentramos refere-se a multiplicidade de soluções distintas, e corretas, para um mesmo problema. Para ilustrar isso, considere o problema 5.1:

Problema 5.1 *Dados dois pontos A e B, construir o ponto médio entre eles.*

Na Figura 5.1 são apresentadas duas construções diferentes que resolvem o problema 5.1. Além destas, uma infinidades de outras construções são possíveis, sendo elas "minimais[2]" ou não. Essa característica em exercícios de Matemática/Geometria dificulta muito o processo de validação automática.

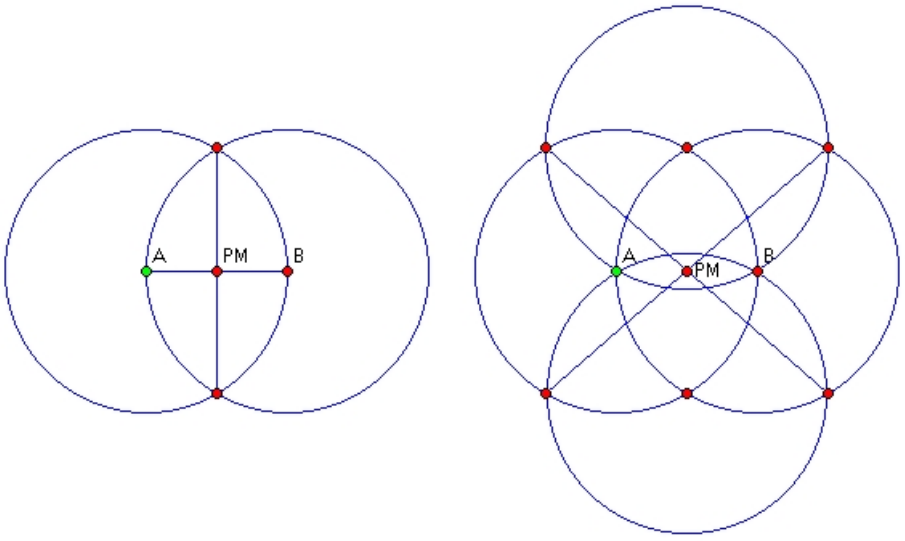

Figura 5.1: Duas construções diferentes do ponto médio

Identificamos duas técnicas básicas para validação automática de exercícios em GD: a **prova automática de teoremas** e a **validação numérica** .

[2]Por construção "minimal", entenda-se aquela na qual nenhum objeto pode ser removido sem comprometer o resultado.

A primeira técnica para validação de exercícios que consideramos foi a prova automática de teoremas. Entretanto optamos por outra técnica, que denominamos validação numérica, por esta atender melhor nossos objetivos.

Além destas, existem outras técnicas para validação/avaliação automática de exercícios que não serão apresentadas nesta dissertação. Uma delas é técnica baseada na teoria ACT-R (versão atual da teoria ACT[3] - *Atomic Component of Thought*), uma teoria geral sobre a cognição humana, que tenta reproduzir a forma como adquirimos conhecimento. Desenvolvida pelo grupo de pesquisa liderado por John R. Anderson da Universidade de Carnegie Mellon, esta teoria é aplicada em programas tutores para ensino de Algebra e de Geometria (Aleven et al., 2004).

Na seção 5.1.1, faremos uma rápida apresentação da prova automática de teoremas e nas seções seguintes nos concentraremos na validação numérica.

5.1.1 Prova Automática de Teorema

Segundo Sutcliffe (2004), a **prova automática de teoremas** (Automatic Theorem Proving - ATP) pode ser resumida em: um programa de computador que mostra se uma sentença (a conjectura) é uma conseqüência lógica de um conjunto de sentenças (os axiomas e hipóteses). A linguagem utilizada pelos programas de ATP deve ser formal de modo a não permitir ambigüidade.

A verificação de uma sentença produzida por um programa de ATP é conhecida como prova. Esta prova descreve uma sequência de passos (conseqüências lógicas) que validam uma conjectura. Os passos seguidos por um programa de ATP (conhecidos também como árvore de prova) podem ser compreendidos e seguidos por outros programas de ATP ou mesmo por uma pessoa. Alguns trabalhos pioneiros no desenvolvimento de provadores automáticos de teoremas de Geometria foram Gelernter (1963), Gelernter et al. (1963) e Gilmore (1970).

Os programas de ATP, em princípio, devem ser capazes de resolver uma infinidade de problemas, gerando diferentes tipos de provas que dependem dos métodos utilizados (Gao & Zhu, 1998; Botana & Valcarce, 2002). Atualmente, existem vários métodos para a prova automática em Geometria, dentre eles segundo (Gao & Zhu, 1999), merecem destaque: o método de Wu, o método de área, a base de Groebner, o método por vetor e o método por ângulos.

Existem alguns programas de geometria dinâmica (GD) que utilizam os métodos apresenta-

[3]Mais detalhes sobre a teoria ACT podem ser encontrados no *site* http://act-r.psy.cmu.edu/

dos acima para fazer a prova automática de teoremas. Dentre eles podemos citar os programas Geolog (Holland, 2002), Geometry Expert (Gao & Zhu, 1998) e o Discover (Botana & Valcarce, 2002).

Na Figura 5.2 é apresentado a árvore de prova criada pelo programa Geolog (Holland, 2002) para verificar se é verdadeira a conjectura *"O ponto E é ponto médio de A e B"* a partir da construção realizada.

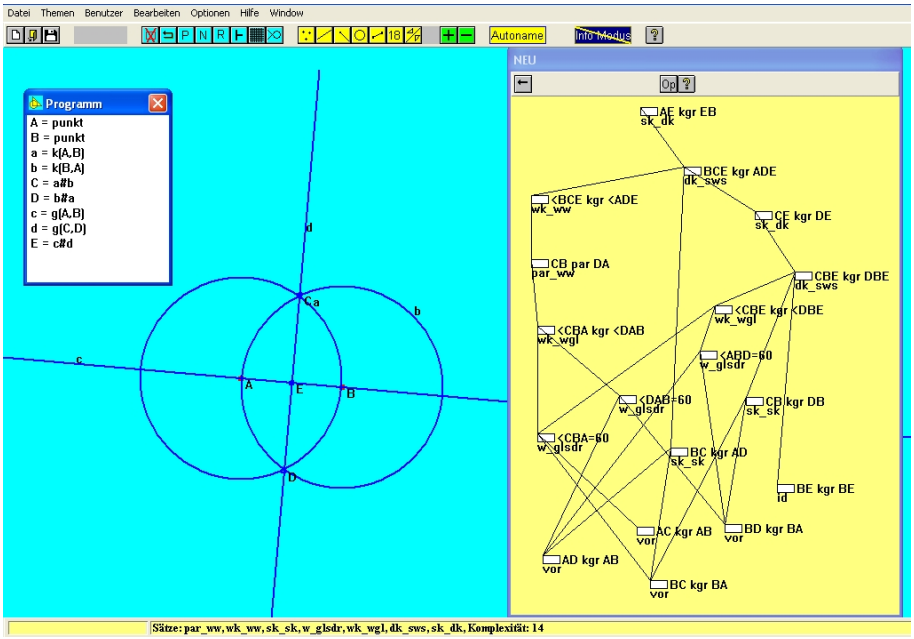

Figura 5.2: Árvore de provaconstruida no programa Geolog

Para Kortenkamp (1999), a Geometria é uma importante área de aplicação da prova automática de teoremas, pois é possível fazer uso de vários métodos para criar provas matemáticas das mais diversas conjecturas geométricas. Contudo, segundo (Gao & Zhu, 1999), devido a complexidade do problema (provar), ainda existem algumas limitações nos métodos de ATP que não permitem a prova automática de qualquer figura geométrica.

Neste trabalho, não utilizaremos o método de ATP para validar os exercícios.

5.1.2 Validação Numérica

Estamos denominando por **validação numérica** o método que compara os objetos-resposta do aluno com os correspondentes do gabarito do professor. Esta comparação é feita a partir de um **critério de distância** entre os objetos. Por exemplo, se o problema é determinar o ponto médio

entre os pontos A e B, e o objeto-resposta do aluno tem o rótulo Ma e o do professor tem rótulo Mp, verifica-se qual a distância entre Ma e Mp, de acordo com o critério estabelecido (no iGeom este critério está definido na seção 5.4.1).

A validação numérica é baseada na estrutura "dinâmica" da GD e parte de um gabarito que o professor deve fornecer, na produção do exercício.

Para evitar (ou ao menos, reduzir) problemas de soluções que funcionem apenas em casos particulares, o processo de validação pode utilizar-se da estrutura "dinâmica" da GD (como no caso do iGeom e do Cinderella): move-se internamente os objetos iniciais (A e B no exemplo do ponto médio) e, para cada configuração, computam-se as distâncias entre as respostas do aluno e do professor. A movimentação não é apresentada na tela e o resultado da validação é definido a partir de um critério sobre todas estas distâncias.

Neste contexto, se existe uma construção conhecida, que é solução para um exercício, e existe outra construção cuja distância entre elas for "aceitável[4]" então podemos "acreditar" que esta outra construção também é solução para o exercício. Como observa Kortenkamp:

> *"A geração de "suficientes" exemplos aleatórios, nos quais as construções mantém suas propriedades, é quase tão convincente quanto uma prova simbólica realizada em computador"*
>
> (Kortenkamp, 1999)

Este método, apesar de não ser uma prova formal como a ATP e exigir uma construção gabarito, possui as vantagens de: (a) utilizar menos processamento computacional; e (b) não restringir o domínio de aplicação, permitindo que um exercício proposto pelo professor possa ser resolvido, utilizando quaisquer técnicas, sem prejudicar o processo de validação.

Com o objetivo de implementar um validador automático no iGeom, que fosse suficientemente rápido em micros caseiros e conseguisse considerar qualquer solução, optamos por implementar a validação numérica.

5.2 Autoria e Validação em Programas de GD

De modo geral, existem várias características importantes em sistemas de apoio ao ensino. Destacamos aqui duas delas, uma relacionada aos professores e outra relacionada aos alunos.

[4]Devido às imprecisões numéricas dos computadores, é necessário aceitar alguma diferença numérica entre as construções.

Para o professor é importante que o sistema simplifique, de alguma forma, seu trabalho. Esta simplificação pode motivá-lo a enfrentar o aprendizado do sistema, que idealmente deve ser de fácil manipulação e, se possível, auto-explicativo.

Em relação ao aluno, além de outras questões didáticas, é importante que ele receba rapidamente a validação de sua solução em cada atividade desenvolvida. Isso permite o maior envolvimento do aluno com o assunto abordado e evita a insatisfação pela demora em obter uma resposta (Hentea et al., 2003).

Nas próximas seções apresentaremos estas duas características (autoria e validação) em três programas de geometria dinâmica, o iGeom, o C.a.R. e o Cinderella, que utilizam a validação numérica para realizar a validação automática de exercícios.

O esquema de autoria de exercício é semelhante nos três programas, possuindo, pelo menos, os seguintes passos:

- Construir o **gabarito**: Uma construção que servirá para fazer a comparação com a solução do aluno em um determinado exercício.
- marcar quais são os **objetos de entrada**: aqueles que o aluno vai receber como enunciado do exercício;
- marcar quais são os **objetos de saída** (resposta): aqueles que serão comparados aos correspondentes da resposta do aluno;
- determinar quais os botões que ficarão disponíveis para o aluno resolver o exercício.

Como a construção do exercício utiliza a mesma área de desenho que será utilizada em sua resolução, o professor pode deixar os objetos nas posições onde gostaria que o aluno os visualizasse, agilizando sua formatação e publicação.

5.2.1 C.a.R.

A autoria de exercícios no C.a.R. possui algumas particularidades que, muitas vezes, acabam dificultando o processo de criação de exercícios. Por exemplo, este programa não possui uma janela própria para auxiliar na autoria de exercícios.

Todos os passos para criar um exercício estão disponíveis em suas opções de menus presentes no programa aplicativo. O processo de autoria envolve oito passos: (a) construção do

gabarito; (b) seleção do último objeto criado; (c) seleção dos objetos-resposta; (d) Esconder objetos (os objetos não escondidos serão as entradas do exercício); (e) Criar enunciado; (f) gravar exercício; (g) Exportar para páginas HTML; (h) definir configuração da página HTML e escolher botões que ficarão visíveis para o usuário.

A validação de exercícios no C.a.R. é tratada como construções incompletas nas quais o usuário deve atingir os objetos-resposta (chamados de objetos-alvo no C.a.R.). Apesar de ser possível testar o exercício no programa aplicativo, este só pode ser efetivamente utilizado em páginas Web, já que o C.a.R. não permite liberar ou bloquear os botões de sua interface quando executado na forma de aplicativo.

O algoritmo de validação automática do C.a.R. considera uma única instância do problema, por exemplo, se o problema for definir o ponto médio dos pontos A e B então o algoritmo verifica a distância entre os pontos Ma, resposta do aluno, e Mp, gabarito do professor, apenas para a instância inicial de A e B. Caso esta distância seja considerada "aceitável" então o programa considera que o ponto Ma está correto.

Para que este algoritmo funcione os pontos A e B não podem ter suas posições modificadas. Apesar desta estratégia funcionar corretamente e permitir que o aluno utilize quaisquer técnicas para resolver o exercício, ela não permite a manipulação dos objetos e, portanto, perde as características e os benefícios da geometria dinâmica. Além disso, com a opção de inserir pontos em coordenadas arbitrárias (recurso disponível no C.a.R.), é possível "enganar" este programa. Por exemplo, no exercício do ponto médio apresentado no parágrafo anterior, o aluno pode inserir um ponto qualquer e digitar as coordenadas da localização do ponto médio. Nesse processo, o algoritmo de validação identifica que o ponto criado possui as mesmas coordenadas do objeto-resposta e retorna que o exercício está correto.

5.2.2 Cinderella

O surgimento do programa de GD Cinderella, em 1999, trouxe duas inovações marcantes: a primeira foi a possibilidade de criar páginas Web e a segunda os recursos de autoria e validação automática de exercícios. Nesta versão de validação automática já era possível que os objetos de entrada do exercício fossem manipulados e, portanto, preservando o "dinamismo".

Comparativamente ao C.a.R., a ferramenta de autoria e validação de exercícios do Cinderella possui mais recursos, por exemplo, permitindo que dicas sejam inseridas quando o aluno não consegue realizar a construção no tempo determinado ou quando este realiza construções pré-

determinadas pelo professor.

O processo para criar um exercício no Cinderella exige no mínimo cinco passos: (a) construção do gabarito; (b) criar enunciado e selecionar os objetos de entrada; (c) selecionar os botões a serem disponibilizados para o usuário; (d) criar mensagens de incentivo e seleção dos objetos-resposta; (e) gravar exercício e exportá-lo para HTML. Caso seja necessário inserir dicas, mais passos intermediários são necessários.

A validação automática desenvolvida no Cinderella foi denominada por seus autores de "verificação automática de teoremas" (*Automatic Theorem Checking*). É um método numérico que gera uma "hipótese", por exemplo, *"um ponto P coincide com outro ponto Q"*, e verifica através de movimentos aleatórios sucessivos se em cada nova posição a "hipótese" se mantém. Segundo Kortenkamp (1999), embora este método não seja uma prova formal, testes empíricos mostraram que este método funciona bem na prática.

Da mesma forma que o C.a.R. o uso da validação automática no Cinderella é possível apenas via Web.

5.2.3 iGeom

O processo para criar um exercício no iGeom possui cinco passos: (a) construção do gabarito; (b) seleção dos objetos de entrada (que inclui a seleção do enunciado); (d) seleção dos objetos-resposta; (c) desabilitação de botões; e finalmente (e) gravar o exercício ou exportá-lo para HTML.

O processo de validação automática no iGeom, de modo semelhante ao Cinderella, é baseado na estrutura dinâmica dos programas de GD e no gabarito do professor. Nosso algoritmo move internamente os objetos da construção e, a cada configuração, anota-se a medida de distância entre a solução do aluno e do professor. Devido às imprecisões numéricas, para considerar em uma determinada configuração que a resposta do aluno equivale à resposta no gabarito, as medidas das distâncias encontradas nas diversas instâncias analisadas devem ser menores do que um valor ε previamente definido. Os detalhes da implementação deste algoritmo serão apresentados na seção 5.4.

5.3 Autoria e Validação no iGeom

Nesta seção apresentaremos os recursos de autoria e validação automática desenvolvidos e implementados no programa iGeom. Apesar de sua similaridade com os programas, C.a.R. e Cinderella, a nossa abordagem permite que tanto o processo de autoria quanto o processo de validação sejam realizados no programa aplicativo ou através de páginas Web. A utilização via Web é útil, principalmente, quando existe um servidor comunicando-se com o iGeom.

O esquema de autoria de um exercício no iGeom começa com a construção geométrica desejada, que servirá como gabarito. Uma vez pronta a construção, utilizando a interface de autoria de exercícios, o professor deve anotar o que o aluno receberá como enunciado e o que deverá ser considerado como resposta. A interface de autoria de exercícios é bastante simples, contando com uma só janela conforma mostra a Figura 5.3.

Figura 5.3: Janela para autoria de exercício no iGeom

Outro recurso do iGeom útil à confecção dos exercícios é a ferramenta para inserir textos. Estes textos podem servir para introduzir comentários ou conter o próprio enunciado do exercício. Este recurso é bastante útil, mesmo que o sistema Web também disponha de uma área específica para expor o enunciado, pois assim armazena-se o enunciado do exercício no mesmo arquivo (facilitando seu uso futuro).

5.3.1 Autoria de Exercícios

O botão que abre as opções de exercício (menu primário) e os botões no menu secundário estão indicados na Figura 5.4.

Figura 5.4: Botões para criação de Exercício

Vamos especificar melhor como deve ser construído um exercício no iGeom. Para isso, utilizaremos o exemplo 2.1, da construção da mediatriz (apresentado na seção 2).

Este exemplo serve para ilustrar que: o método de validação implementado não exige que o aluno use a mesma técnica de solução adotada para construir o gabarito e que o professor pode utilizar, na construção do gabarito, ferramentas que não ficarão disponíveis para o aluno.

1. **Construir o gabarito**
   A construção do gabarito é realizada como qualquer outra construção. Para o exemplo, devemos fazer a construção da mediatriz de dois pontos, criando os pontos A e B, o segmento $s0$, o ponto médio M e a reta r (mediatriz) perpendicular ao segmento $s0$ passando por M (Figura 5.5)

2. **Anotar os objetos de entrada**
   Abrir a janela de autoria de exercícios clicando primeiro no botão de "exercícios"  , no menu principal, e depois no botão "Criar Exercícios"  , no menu secundário.
   Na janela que será aberta (vide Figura 5.3), marcar os objetos de entrada, usando o botão "marcador".  Com este botão selecionado, clicar nos pontos A e B e também no texto criado como enunciado: *"Dados dois pontos, A e B, construir sua mediatriz"*
   Dentro da janela de autoria de exercício, clicar no botão "Inserir Seleção" ao lado da área de "Objetos Escolhidos como Entrada" (área de texto mais acima da janela de autoria).

3. **Anotar os objetos de resposta (saídas)**
   Marcar os objetos de saída de modo análogo ao feito com os objetos de entrada: com o

botão "marcador" selecionado, clicar na reta r correspondente à mediatriz.
Dentro da janela de autoria de exercício, clicar no botão "Inserir Seleção" ao lado da área de "Objetos Escolhidos como Resposta" (área de texto mais abaixo da janela).

4. **Selecionar os botões a serem disponibilizados ao aluno**
   Ainda na janela de autoria de exercícios, pode-se clicar em todos os botões que não devem ser utilizados na resolução do exercício.
   Neste exemplo, pode-se eliminar os botões: *Ponto Médio*, *Perpendicular* e *Paralela*, para que o aluno não utilize estas ferramentas.

5. **Gravação/envio do exercício**
   Clicar no botão *OK* da janela de autoria de exercício. Ficarão visíveis na área de desenho apenas os objetos de entrada, sendo esta a forma que o aluno receberá o exercício.
   Existem duas possibilidades de uso, uma via aplicativo e outra via Web, conectando-se a um servidor. Quando usada via Web, após o clique no botão OK (da janela de autoria), o gabarito é enviado ao servidor (para este armazená-lo).
   Quando a construção do exercício for via aplicativo, abrem-se mais três possibilidades de gravação:

   - formato para publicação em páginas Web (legível pela versão *applet* e aplicativo do iGeom);
   - formato para publicação em Cursos na Web (legível apenas via *applet*);
   - formato para ser utilizado no iGeom aplicativo (legível apenas via aplicativo).

Na Figura 5.5, sobre a janela principal do iGeom, aparece a janela de autoria de exercícios, com os objetos de entrada e os objetos-resposta selecionados.

5.3.2 Validação de Exercícios

Associado ao recurso de autoria de exercício, foi também desenvolvido um recurso para validação automática da solução do aluno. A técnica implementada para realizar esta validação é fortemente baseada na estrutura "dinâmica" da GD e no gabarito que o professor fornece ao gerar um exercício.

Quando o aluno abre uma página com o exercício produzido no exemplo da mediatriz, apresentado na seção 5.3.1, verá apenas os pontos A e B e a mensagem: *"Dados dois pontos, A e B,*

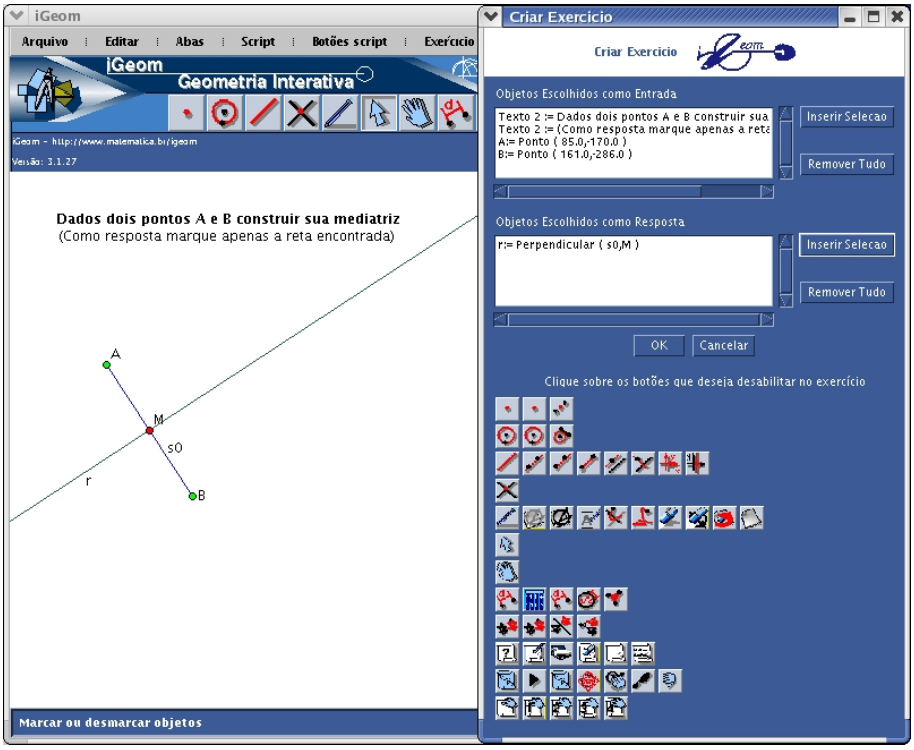

Figura 5.5: Construção do gabarito para o exercício da mediatriz

construir sua mediatriz". Além disto, os botões que o professor selecionou na janela de autoria não aparecerão no menu de botões do iGeom, seja em modo *applet* ou aplicativo (Figuras 5.6 e 5.7).

Figura 5.6: Opções de reta

Figura 5.7: Opções de reta sem alguns botões

Após terminar sua construção (Figura 5.8) o aluno deve selecionar, com o botão "marcador", o objeto (ou objetos) que supõe resolver o problema e clicar no botão de envio de resposta  . Ao executar esta operação, o algoritmo de validação automática será iniciado. Caso o aluno selecione algum objeto diferente daqueles existentes no gabarito, receberá uma mensagem de erro específica para cada caso. Em não havendo este tipo de erro, o algoritmo de validação será disparado devolvendo: correto ou incorreto. Nos casos em que o algoritmo devolver incorreto, o iGeom poderá mostrar um contra-exemplo: uma configuração da construção em que fica óbvio o erro do aluno (de modo mais formal: as distâncias entre suas respostas e as respostas do

gabarito do professor são maiores que um limite de tolerância).

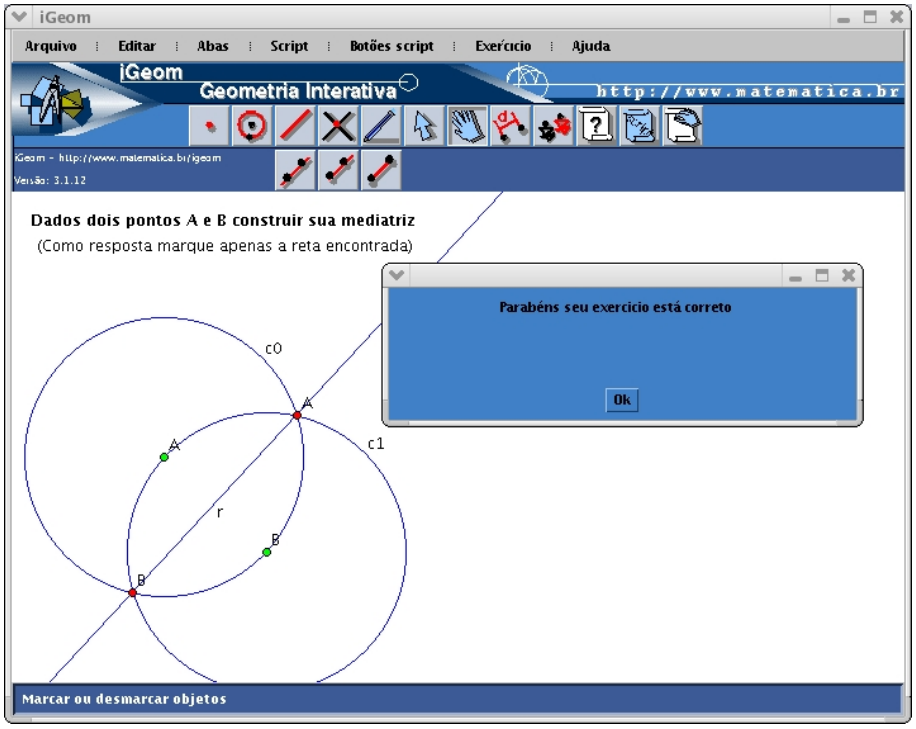

Figura 5.8: Construção do aluno para o exercício da mediatriz e o resultado da validação

O resultados poderá ser apresentados ao usuário para notificá-lo do resultado da validação de seu exercício.

5.4 O Algoritmo de Validação Automática no iGeom

Resumidamente, os objetivos do validador automático de exercícios são:

1. Detectar quaisquer solução correta, mesmo aquelas que utilizam técnicas diferentes das imaginadas pelo professor;

2. Apresentar de imediato o resultado da validação;

3. Fornecer um “contra-exemplo” se a resposta do aluno estiver errada.

5.4.1 Validação Numérica

A linha de trabalho que adotamos é fortemente baseada no gabarito do professor, que deve ser não ambíguo (veja seção 5.5). O resultado da validação é uma medida de distância entre a solução do aluno e o gabarito do professor. Como a distância é obtida a partir da descrição numérica dos objetos, denotaremos esta por **validação numérica** .

Para se fazer esta validação é necessário definir o critério de **distância** entre os pares de objetos geométricos. Definimos o critério de distância apenas para pares de objetos de mesma família (ou tipo). Para simplificar, vamos nomear apenas alguns dos exemplos de famílias mais comuns numa construção: família dos pontos (F_p); família das circunferências (F_c); e família dos segmentos (F_s). O conjunto de todas as famílias de objetos será representado por F_{og}.

Deste modo, o critério de distância pode ser uma função $dist$ que recebe um par de objetos geométricos $(og1, og2) \in F_{og} \times F_{og}$ e devolve um valor em $\mathcal{R}_+$:

$$dist : (og1, og2) \longrightarrow \mathcal{R}_+. \tag{5.1}$$

A descrição computacional dos objetos é definida para cada configuração da construção e deste modo pode ser feita a partir de uma lista de valores numéricos. Por exemplo: um ponto pode ser representado por um par (x, y), onde x e y são as coordenadas do ponto; uma circunferência pode ser representada por uma tripla (x, y, r), sendo (x, y) as coordenadas de seu centro e r seu raio; e um segmento $s = [(x_1, y_1), (x_2, y_2)]$, pode ser representado pela quádrupla (x_1, y_1, x_2, y_2). Deste modo, se considerarmos apenas as famílias de pontos, circunferências e segmentos, podemos definir $dist$ conforme a equação 5.2. Dados dois objetos quaisquer de mesma família, $og1$ e $og2$, se $(l_1^1, l_2^1, \ldots, l_{i_1}^1)$ e $(l_1^2, l_2^2, \ldots, l_{i_2}^2)$ são as listas que representam, respectivamente, $og1$ e $og2$, então:

$$dist(og1, og2)^5 = \begin{cases} |l_1^1 - l_1^2| + |l_2^1 - l_2^2| & , (og1, og2) \in F_p \times F_p \\ |l_1^1 - l_1^2| + |l_2^1 - l_2^2| + |l_3^1 - l_3^2| & , (og1, og2) \in F_c \times F_c \\ \min \left\{ \begin{array}{l} \sum_{i=1}^{4} |l_i^1 - l_i^2|, \\ \sum_{i=1}^{4} |l_i^1 - l_{(i+1)\%4+1}^2| \end{array} \right\} & , (og1, og2) \in F_s \times F_s. \end{cases} \tag{5.2}$$

A necessidade do mínimo (min) quando os objetos forem do tipo segmento é devido ao desejo de classificar como iguais os segmentos AB e CD, mas também suas permutações AB e

[5] O símbolo $\%$ está sendo empregado como o resto da divisão inteira (função módulo).

DC, BD e CD e BA e DC. Ou seja, se for segmento, o critério de distância não distingue um segmento AB do segmento BA.

Uma vez definida a distância entre objetos, podemos definir a distância entre pares de construções distintas a partir de seus objetos. Sendo OG_p e OG_a, duas construções, e seus objetos representados, respectivamente, por $(og_1^p, og_2^p \ldots og_i^p)$ e $(og_1^a, og_2^a \ldots og_j^a)$, a distância entre OG_p e OG_a pode ser expressa conforme a tabela 5.1. Sendo $tipo(og)$ uma função que devolve o tipo do objeto geométrico og.

Tabela 5.1: Definição da distância entre pares de construções

Se a cardinalidade das listas OG_p e OG_a ($\#OG_p$ e $\#OG_a$) forem distintas, então
$\boldsymbol{dist(OG_p,\ OG_a) = +\infty}$.

Se $\#OG_p = \#OG_a = n$,
sejam I_p = $(p_1, \ldots, p_n)$ e $I_a = (a_1, \ldots, a_n)$ duas permutações sobre os n primeiros naturais,

$$dist(I_p, I_a) = \left\{ \begin{array}{ll} \displaystyle\sum_{i=1}^{n} dist(og_{p_i}^p, og_{a_i}^a) & \text{, se } \left\langle\ tipo(og_{p_i}^p) = tipo(og_{a_i}^a), i \in \{1, \ldots, n\}\ \right\rangle \\ +\infty & \text{, c.c.} \end{array} \right\}$$

então
se $\mathcal{P}_n$ é o conjunto de todas as permutações dos n primeiros naturais,

$$\boldsymbol{dist(OG_p,\ OG_a) = min\left\{dist(I_p, I_a), \forall (I_p, I_a) \in P_n \times P_n\right\}}$$

ou seja, dentre todas as permutações de objetos de OG_p e OG_a, $dist(OG_p,\ OG_a)$ é a soma das distâncias entre cada par de objetos de mesmo tipo que resulta no menor valor.

Esta é apenas uma das possibilidades para medir distância entre construções. Uma generalização simples desta função é colocar coeficientes positivos para ponderar cada objeto.

Uma **solução** (construção geométrica) pode ser representada como uma **função** que recebe uma lista de objetos geométricos (**entrada**) e devolve uma outra lista de objetos geométricos (**saída**), que podemos representar como:

$$S : OG_i \longrightarrow OG_f. \tag{5.3}$$

Uma **instância** de uma construção S é a aplicação de S sobre uma dada configuração de

objetos. Uma vez determinada a distância entre pares de listas de objetos e a representação de uma solução, podemos definir quando duas construções (soluções) são equivalentes, como segue 5.2.

Definição 5.2 (Equivalência) *Sejam S_p e S_a duas construções aplicáveis sobre a mesma lista de objetos geométricos OG. Então S_a e S_p são equivalentes se, e somente se, para qualquer configuração OG_0 da lista OG, tivermos $dist(S_p(OG_0), S_a(OG_0)) = 0$.*

Vale observar que se duas construções são equivalentes, a distância entre ambas é invariante em relação às configurações iniciais, isto é, à distância computada para qualquer instância é sempre nula. Portanto, se desconsiderarmos erros numéricos, um bom critério de validação é dizer que uma construção S_a esta correta sempre que for equivalente à construção S_p gabarito. Entretanto existe o problema prático: *como implementar uma versão suficientemente rápida e levando em consideração os erros numéricos ?*

Como construções distintas que obtém os mesmos objetos-resposta podem resultar em pequenas diferenças numéricas, optamos por relaxar o critério de equivalência, permitindo que, por exemplo, dois pontos "muito próximos" em um bom número de distintas instâncias sejam considerados "quase equivalentes" (vide seção 5.4.2). Esta solução, em princípio, permite que sejam construídos exemplos em que ocorram erros de ***falso positivo*** (exercício errado avaliado como correto) ou de ***falso negativo*** (exercício certo avaliado como incorreto), mas funciona bem na prática. Também é possível aumentar o número de instâncias a serem testadas ou fazer outras modificações de parâmetros para reduzir/eliminar validações falsas.

Nas próximas seções, quando estiver implícito, ou for indiferente qual a lista de objetos geométricos OG, utilizaremos a notação simplificada S e não $S(OG)$.

5.4.2 O Algoritmo Validador

Baseado na validação numérica apresentada na seção 5.4.1, implementamos no iGeom um algoritmo validador composto de quatro passos principais: transformação numérica, análise, instanciação e validação. Como apresentado na seção 5.3, para efetuar a validação é necessário que o professor anote durante a construção do gabarito quais objetos serão avaliados (**objetos-resposta**). Ao final de sua resolução, o aluno deve anotar quais são seus objetos-resposta,

que devem ser iguais aos do gabarito em número e tipos, e depois solicita a validação[6] da sua solução.

As listas de objetos-resposta do professor e do aluno serão os dados de entrada do validador e o resultado será um natural entre 1 e 3, mas este intervalo de resultados é facilmente alterável. Nas duas sub-seções seguintes detalharemos este algoritmo.

Transformação e Análise

A **transformação numérica** quando aplicada sobre um objeto geométrico, devolve uma lista de escalares que representam este objeto. Na maioria dos casos esta transformação é simples como apresentado na seção 5.4.1, porém alguns objetos, como o polígono, necessitam além da transformação numérica, a ordenação dos pontos que o representam. Com esta lista de escalares podemos fazer a comparação entre objetos. Assim, para analisar duas construções, o gabarito S_p do professor e a solução S_a do aluno, transformamos os objetos marcados como resposta em listas de escalares para então fazer a avaliação, como esquematizado na Figura 5.9.

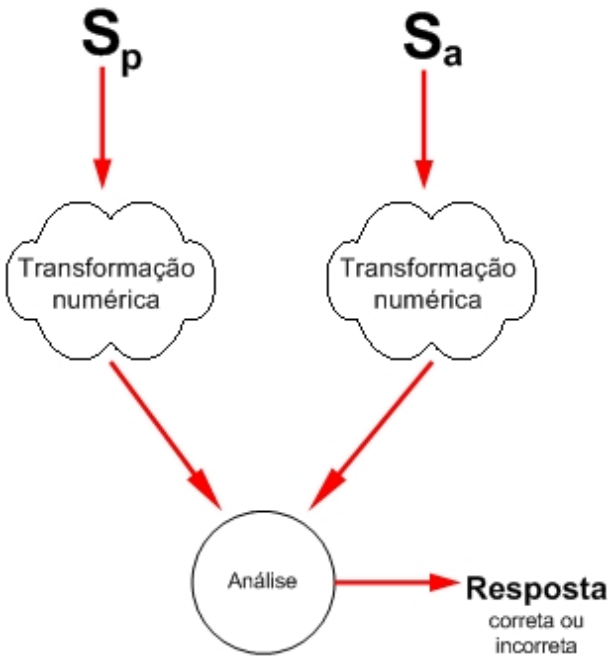

Figura 5.9: Transformação numérica e análise

[6]No caso de uso do *iGeom* em um sistema fechado, com um servidor e alunos registrados, o professor pode optar por não mostrar ao aluno/internauta o resultado da validação e apenas arquivar no servidor a solução.

A **análise** é feita em duas etapas. A primeira etapa consiste no mapeamento entre as listas e a segunda na comparação delas através do critério de distância. A razão da primeira etapa é permitir que o aluno tenha a liberdade de fazer a marcação dos objetos-resposta em qualquer ordem. Por exemplo, caso a solução do professor seja representada pelos pontos A e B e a do aluno pelos pontos C e D, a etapa de mapeamento identificará se o ponto C corresponde a A ou a B, fazendo o mesmo para D.

O **mapeamento**[7] dos objetos de S_a e S_p é realizado comparando-se cada elemento og de S_a com todos os elementos de S_p que pertençam a mesma família de og e minimizando a distância entre eles. Assim, um objeto de S_a será mapeado em um objeto de S_p se ambos pertencerem à mesma família de objetos geométricos, ainda não foram mapeados e a distância entre eles for a menor possível em relação aos outros objetos de S_a. Este mapeamento é feito apenas para a primeira instância (a configuração inicial do enunciado do exercício). A segunda etapa da avaliação consiste na **comparação** entre os pares de objetos geométricos mapeados (og_i^a, og_i^p). Nesta comparação, utilizamos uma versão relaxada do conceito de equivalência expresso na definição 5.2 (seção 5.4.1), para levar em consideração as imprecisões numéricas ao se empregar diferentes soluções. Assim, adotamos uma margem de erro ε.

Definição 5.3 (Quase Equivalência) *Fixado um conjunto de entradas OG, seja S_p uma construção sobre OG e S_a outra construção sobre o mesmo OG. Então S_a e S_p são quase equivalentes se, e somente se, para qualquer configuração OG_0 da lista OG, tivermos $dist(S_p(OG_0), S_a(OG_0)) < \varepsilon$.*

De modo simplificado apresentamos o psedo-código do validador na tabela 5.2. As linhas 1 a 12 são referentes ao mapeamento dos objetos e as linhas 13 a 19 são referentes a análise (verificando se cada menor distância entre pares de objetos de mesmo tipo é menor do que a tolerância de erro ε).

Note que esta parte do algoritmo trata apenas uma instância da solução, considerando uma posição fixa para cada objeto de entrada. Se utilizarmos apenas esta validação da configuração inicial, pode ocorrer, com freqüencia, erros de falso positivo. Por exemplo, no problema do ponto médio o aluno poderia tentar colocar um ponto "solto" sobre o segmento AB e movê-lo de modo a ficar próximo à posição do ponto médio, sem efetuar uma construção geométrica válida. Apesar de ser difícil posicionar o ponto de modo que o algoritmo avalie a solução como

[7] Se os objetos coincidirem em número e tipo será uma função bijetora.

Tabela 5.2: Pseudo-código do validador implementado no iGeom

1 . Recebe duas listas S_a e S_p de objetos geométricos
2 . Crie uma nova lista de objetos geométricos $S_t \leftarrow \emptyset$
3 . Para cada elemento og_i^p da lista S_p
4 . MenorDistanciaEncontrada $\leftarrow \infty$
5 . ObjetoCorrespondente $\leftarrow \emptyset$
6 . Para cada elemento og_j^a da lista S_a
7 . Se og_i^p e og_j^a pertencem a mesma famílias de objetos geométricos então
8 . Se dist(og_i^p, og_j^a) < MenorDistanciaEncontrada
9 . ObjetoCorrespodente $\leftarrow og_j^a$
10. MenorDistanciaEncontrada $\leftarrow$ dist(og_i^p, og_j^a)
11. Se ObjetoCorrespondente = $\emptyset$
12. Devolva Falso
13. Senão
14. $S_t \leftarrow S_t \bullet$ ObjetoCorrespodente //concatenação
15. Remove ObjetoCorrespondente de S_a //objeto está mapeado
16. Para cada elemento og_i^p da lista S_p
17. Seja og_j^t o primeiro elemento S_t
18. Se dist(og_i^p, og_j^t) < ε então
19. Remova og_j^t de S_t
20. Senão
21. Devolva Falso
22. Devolva Verdadeiro

correta, é possível conseguir isso (este erro aparece no algoritmo avaliador do C.a.R.). Outro exemplo que ilustra este erro de falso positivo é apresentado no problema 5.4.

Problema 5.4 *Dado dois pontos A e B, construir um triângulo eqüilátero $\triangle ABC$.*

Para resolver este problema são apresentadas duas soluções (Figura 5.10 e 5.11). A primeira representa uma construção correta para o problema. A segunda é a construção de um triângulo isósceles, com o ponto C solto sobre a reta s, mas coincidentemente nesta configuração da construção tem o ponto C na posição tal que $\overline{AB} = \overline{AC} = \overline{BC}$. Neste caso, a construção do triângulo isósceles foi erroneamente utilizada para se produzir um triângulo eqüilátero. Porém, esta "quase equivalência" só se verifica para a configuração inicial. Ao movermos todos os pontos "soltos[8]" da construção, em particular o ponto C do triângulo isósceles, ficará claro que $\triangle$ABC não é equilátero.

[8]Um ponto é considerado solto quando é possível manipulá-lo (modificar sua posição) diretamente. Por exemplo, através da opção "mover ponto" presente no menu de botões é possível modificar um ponto livre na tela ou um ponto localizado em cima de outro objeto. Pontos como o ponto de interseção, ponto médio e pontos de isometrias não são manipulados diretamente.

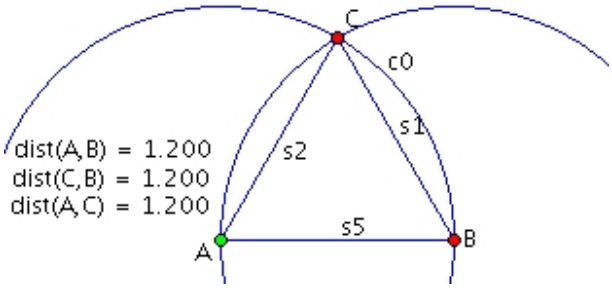

Figura 5.10: Construção do △ eqüilátero.

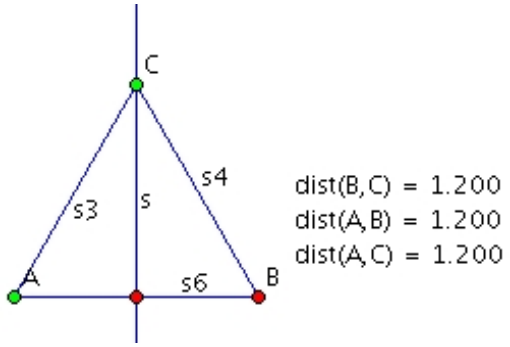

Figura 5.11: Construção do △ isósceles com $\overline{AB} = \overline{AC} = \overline{BC}$.

Instanciação e Validação

Como já foi observado, uma maneira simples de detectar os erros apontados no final da seção 5.4.2 é criar um mecanismo que analise o exercício em várias instâncias (instanciação) e somente após um número considerável de validações o sistema devolve o resultado final da validação . A simulação do procedimento que utiliza em diversas instâncias o algoritmo apresentado na seção 5.4.2 pode ser visualizada na Figura 5.12.

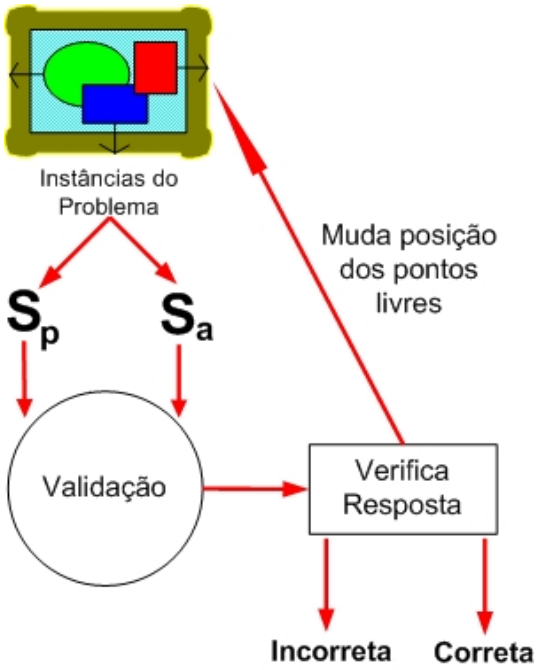

Figura 5.12: Validação Automática

A cada iteração do algoritmo de validação automática (Figura 5.12) os objetos geométricos devem ter suas posições alteradas. Essa alteração é feita através da movimentação "aleatória" de todos os pontos "soltos" da construção. A escolha da nova posição de um ponto é feita modificando suas coordenadas (x, y) para (x', y') , $x - k < x' < x + k$ e $y - k < x' < y + k$, onde k é um valor aleatório tal que $0 < k \leq 20$. O valor de k foi determinado empíricamente e permite que a modificação nas coordenadas sejam suficientes para identificar possíveis problemas em uma construção.

Na Figura 5.13, apresentamos a construção incorreta do ponto médio. Ao modificarmos as coordenadas (x, y) do ponto B para $(x + 10, y + 10)$, observamos que o ponto D "descolou-se" de $c1$ (ou seja, D não é um ponto de interseção entre $c0$ e $c1$).

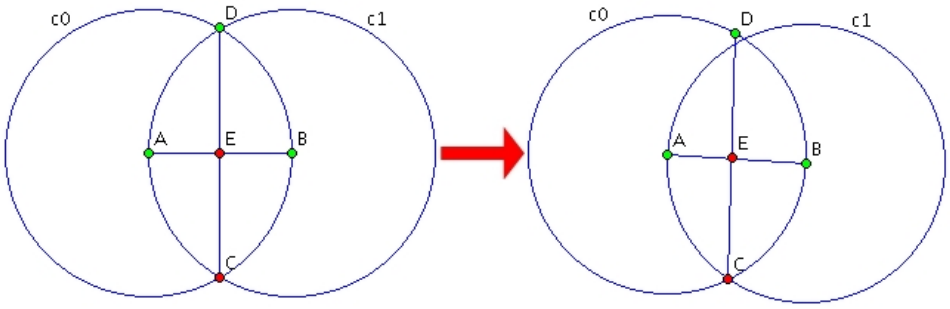

Figura 5.13: Construção incorreta do ponto médio identificada ao movimentar o ponto B

Ao final destes passos, retornamos um valor inteiro entre 1 e 3: (1) correto, não encontrou nenhum contra-exemplo; (2) incorreto, porém encontrou instâncias consideradas corretas e outras incorretas; (3) incorreto, encontrou apenas instâncias incorretas.

5.5 Identificação de Ambiguidade

Como apontado anteriormente, consideramos que uma construção é uma função que recebe um conjunto de objetos de entrada e devolve um conjunto de objetos resposta (representação 5.3). Esta função é considerada **ambígua** quando não é bijetora, ou seja, quando possuir mais de um conjunto de resposta para um mesmo conjunto de entrada. Na GD, isso pode ocorrer se for utilizado algum ponto "solto". Um exemplo é apresentado no problema 5.5.

Problema 5.5 *Dado dois pontos A e B, construir um triângulo isósceles* $\triangle ABC$.

Observe que este problema é intrinsecamente ambíguo, possuindo infinitas soluções, pois o ponto C pode localiza-se em qualquer posição da reta mediatriz de A e B. Dessa forma, independentemente de A e B, o ponto C pode ser movido, modificando a resposta (Figura 5.14).

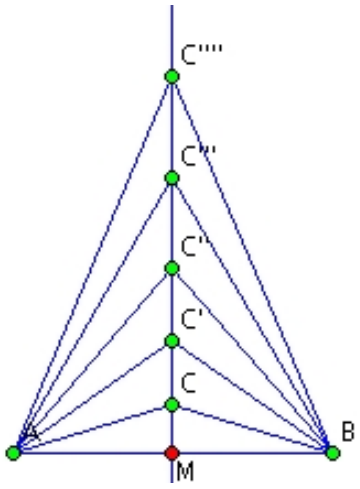

Figura 5.14: Para uma mesma posição dos pontos A e B, temos potencialmente infinitas soluções para o problema 5.5

Com o intuito de identificar este problema de ambiguidade implementamos um algoritmo que faz esta identificação durante a construção do gabarito, avisando o usuário qual é o objeto que torna a construção ambígua (no caso do problema acima seria o ponto C). Este aviso é muito útil para corrigir possíveis distrações do professor durante a criação de um exercício. Os programas C.a.R. e Cinderella, não possuem algoritmo semelhante.

Quando o professor faz a construção do gabarito e seleciona os objetos de entrada e de resposta, o algoritmo de identificação de ambiguidade verifica se existe algum objeto-resposta dependente de algum ponto P que não pertence aos objetos de entrada e nem é determinado unicamente por eles. Caso exista, verificamos ao movimentar este ponto P se a posição de algum objeto-resposta se altera.

No exemplo do triângulo $\triangle ABC$ isósceles, caso um professor crie um gabarito selecionando os pontos A e B como entradas e os segmentos $\overline{AC} e \overline{BC}$ como resposta, o algoritmo identifica a ambiguidade devolvendo uma mensagem ao professor avisando que o ponto C precisa ser selecionado como entrada para remover a ambiguidade do gabarito.

Vale notar que um objeto-resposta que depende de um ponto “solto” não pertencente às entradas do exercício, nem sempre terá sua posição modificada quando o ponto ”solto” é movido.

Um exemplo é a construção de um gabarito para o problema 5.6.

Problema 5.6 *Dado a reta r e o ponto A, construir a reta s que passa por A formando um ângulo de 60 graus à r.*

O gabarito para este problema pode ser visto na Figura 5.15. Os objetos selecionados como entrada são a reta r e o ponto A, e o objeto selecionado como resposta é a reta s. Observe na construção da Figura 5.15 que a reta s é dependente do ponto C. Este ponto não foi selecionado como entrada, contudo ao movimentá-lo a posição da reta não se modifica e, portanto, o gabarito não é ambíguo (Figura 5.16).

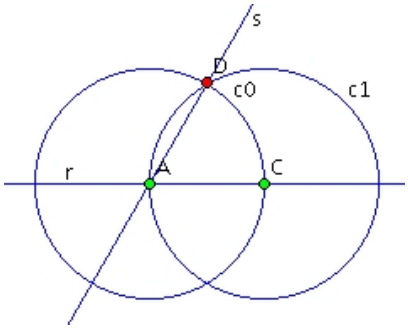

Figura 5.15: Gabarito para a o problema 5.6

O algoritmo desenvolvido para detectar este caso utiliza parte do mecanismo da validação automática apresentada na seção 5.4, fazendo a comparação entre a construção na configuração inicial, com a construção na configuração após a movimentação dos objetos.

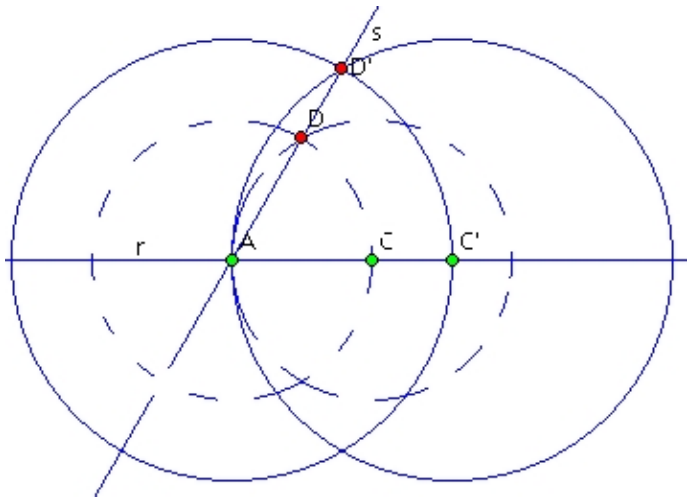

Figura 5.16: Movimentação do ponto C para a posição C'

5.5.1 Consideracões Didáticas Sobre a Geração de Gabaritos

Além do problema da ambiguidade na construção, também existe o problema de ambiguidade no enuncidado. Por exemplo, um exercício proposto pelo professor poderia ter o enunciado: *"Dados dois ponto A e B, construir um triângulo equilátero"* . Como mostram as Figuras 5.17

e 5.18, existem duas soluções possíveis: construir o triângulo equilátero utilizando a interseção norte N, ou a interseção sul S, entre as circunferências.

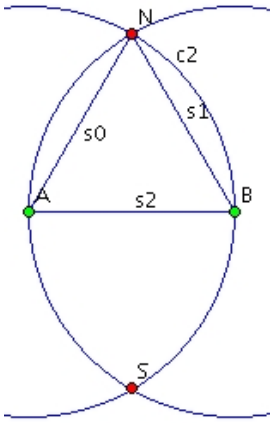

Figura 5.17: Construção do △ eqüilátero utilizando a interseção superior

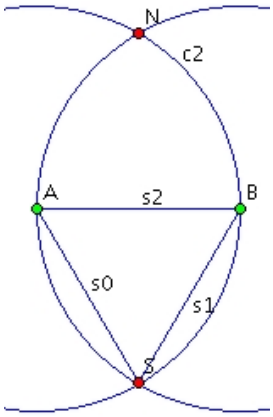

Figura 5.18: Construção do △ eqüilátero utilizando a interseção inferior

Devido a estas particularidades, o professor deve ter um cuidado maior em relação aos enunciados dos exercícios propostos. Segundo os dados obtidos durante o uso do iGeom no curso de MAC118, oferecido no primeiro semestre de 2004, cerca de 69% dos alunos que tiveram seus exercícios invalidados (avaliados como incorretos) tiveram dificuldade para interpretar o enunciado.

Capítulo 6

Conclusões

A possibilidade de oferecer, sem restrições, todos os benefícios e recursos de um programa de GD na Web tem sido para nós um grande desafio e uma forte motivação para o desenvolvimento e implementação de recursos facilitadores ao ensino-aprendizagem no iGeom. Neste contexto, os recursos de comunicação, autoria e validação automática de exercícios desenvolvidos no presente trabalho, visaram auxiliar tanto o professor quanto ao aluno.

Para o professor, oferecemos recursos para criar e validar automaticamente os exercícios e suas utilizações em cursos fechados ou abertos na Web. Dessa forma, reduzimos a carga de trabalho do professor em criar e validar os exercícios realizados por seus alunos e provemos recursos para catalogar as soluções de cada aluno. Assim, ajudamos a sanar um dos grandes problemas apresentado por Bellemain (2002): *a dificuldade do professor em acompanhar o aluno e validar suas construções durante as atividades com os programas de GD*.

Para o aluno proporcionamos o uso das ferramentas da GD diretamente em páginas Web e, através do recurso de validação automática, podemos oferecer respostas rápidas para cada exercício realizado. Com isso, contribuímos para que o aluno tire suas dúvidas imediatamente após o surgimento das mesmas e diminuímos o sentimento de frustração do aluno pela falta de uma resposta imediata.

O programa de GD, apresentado nesta dissertação, o iGeom, permite o uso de todos os seus recursos em ambientes presenciais ou pela Internet. Além disso, utilizando as ferramentas implementadas neste trabalho foi possível integrar o iGeom em um sistema gerenciador de cursos na Web (SAW) para o oferecimento de cursos de Geometria à distância. Estas ferramentas foram testadas em diversas ocasiões: No primeiro semestre de 2004 na disciplina MAC118 (*http://www.ime.usp.br/l̃eo/mac118/04*), contando com três turmas e mais de 150 alunos. No

início de 2005, nos cursos de verão do IME-USP, os *"Cursos do Laboratório de Ensino de Matemática"* e o curso de *"Construção de cursos à distância de geometria: para professores de matemática"*, que contaram com a participação de mais de 25 professores do ensino fundamental e médio.

Os resultados desta dissertação podem ser conferidos na atual versão do *iGeom*, disponível gratuitamente no endereço *http://www.matematica.br/igeom*.

6.1 Contribuições

O presente trabalho descreve o desenvolvimento dos principais recursos implementados no programa iGeom durante o período de 2003 à 2005, dando destaque à sua utilização em ambientes Web. Na tabela 6.1, apresentada também na seção 3.3, destacamos alguns dos recursos existentes no iGeom em comparação aos programas de GD apresentados na seção 2.3.

Tabela 6.1: Recursos de alguns dos programas de GD

Programa	Portável	ADDs	Script	Rec	Web	AA	Com	Licença
iGeom	X	X	X	X	X	X	X	Gratuito
Cabri		X	X					Comercial
C.a.R.	X		X		X	X		GNU
Cinderella	X		X		X	X		Comercial
GSP		X	X	X				Comercial
Tabulae	X	X			X		X	Comercial

Na tabela 6.1, a coluna "Portável" refere-se aos programas que podem ser executados em qualquer plataforma (atualmente restringindo-se àqueles implementados em Java). A coluna "ADDs" refere-se àqueles que permitem a abertura de múltiplas áreas de desenho. Na coluna "Script" encontram-se aqueles que possuem recursos para a criação de *scripts* (macros). Na coluna "Rec" encontram-se aqueles que permitem a criação de *scripts* recorrentes. Na coluna "Web" estão aqueles que permitem o uso irrestrito de seus recursos diretamente em páginas Web. Na coluna "AA" estão aqueles que possuem recursos para autoria e validação automática de exercícios. Na coluna "Com" estão àqueles que possuem recursos de comunicação. E finalmente, a coluna "Licença" refere-se ao tipo de licença que cada um dos programas possui.

Como resultados deste trabalho, apresentamos as seguintes contribuições:

- No trabalho Brandão & Isotani (2003), apresentamos o papel de destaque que a Geometria

Dinâmica tem adquirido no contexto do ensino de Matemática. Fizemos também a comparação do iGeom com alguns dos principais programas de GD. E finalmente, discutimos como soluções geométricas podem ser vistas como algoritmos e como é possível implementar algoritmos geométricos com laços repetitivos, de modo automático, no iGeom.

- Nos trabalhos Brandão et al. (2004b) e Brandão et al. (2004c), apresentamos a integração do iGeom no SAW, um sistema gerenciador de cursos pela Web também em desenvolvimento no IME-USP.

- Nos trabalhos Isotani & Brandão (2004a), Isotani & Brandão (2004b) e Isotani & Brandão (2005), discutimos a implementação dos recursos para autoria e validação automática de exercícios no iGeom, que viabilizaram a avaliação imediata do tipo correto/incorreto e a autoria de exercícios diretamente pela Web. Além disso, apresentamos alguns resultados positivos, que foram obtidos com a utilização destes recursos em um sistema gerenciador de cursos pela Web, o SAW.

- Os recursos desenvolvidos e sua utilização pela Web foram apresentados em duas mostras de *software* (Brandão & Isotani, 2004; Isotani & Brandão, 2004c);

- Os resultados apresentados deram oportunidade para o oferecimento de um mini-curso: *"Geometria Dinâmica com o iGeom: algoritmos geométricos, autoria e avaliação automática de exercícios"*, para capacitar professores e alunos no uso do iGeom pela Web. Este mini-curso foi realizado no Simpósio Brasileiro de Informática na Educação (SBIE) e o texto produzido incluído no livro de Mini-cursos deste evento (Brandão et al., 2004a).

6.2 Trabalhos Futuros e em Andamento

O iGeom é um programa de GD em contínuo desenvolvimento. Os trabalhos aqui reportados são apenas algumas das ferramentas que este programa incorporou nos últimos dois anos. Dentre as ferramentas já em devolvimento utilizando alguns dos recursos implementados neste trabalho são: a animação interativa e a cooperação.

Utilizando os recursos implementados neste trabalho, pretendemos estender o processo de interação dos alunos com os exercícios utilizando a **animação**. Dessa forma, será possível que o professor desenvolva animações (criação e movimentação automática de objetos geométricos na janela do iGeom) que utilizam as construções dos alunos como parte de uma animação.

A cada passo da animação será possível pedir que o aluno resolva um problema e, utilizando o algoritmo de validação automática verificarmos se a solução está correta, e caso a resposta seja afirmativa, utilizamos esta solução (ou pelo menos parte dela) para dar continuidade às animações seguintes.

O **aprendizado cooperativo** é uma técnica muito utilizada em EAD. Seu objetivo é permitir o desenvolvimento de atividades em grupo e proporcionar a cooperação mútua entre seus participantes. Estamos interessados em dotar o iGeom de recursos de cooperação e com isso ampliar seu potencial educacional. Para tanto, queremos desenvolver espaços compartilhados (área de desenhos públicas) para permitir que grupos de alunos compartilhem construções de modo síncrono. Essa extensão implica no desenvolvimento e definição de um protocolo de controle para definir, entre outras coisas, a maneira de se criar e remover objetos. Embora o uso do iGeom de forma cooperativa ainda esteja em estudo, contudo podemos identificar algumas possibilidades no controle dos objetos. Por exemplo, podemos imaginar o controle onde apenas um usuário pode criar e manipular os objetos, ou seja os objetos possuem dono. Outra forma de controle, seria liberar qualquer usuário para criar e manipular os objetos, porém apenas um único usuário poderia fazê-lo por vez.

Além destas novas ferramentas, os desafios futuros que pretendemos vencer são: a incorporação de recursos para melhorar a criação de exercícios e a resposta (*feedback*) dada ao aluno; e a criação de uma biblioteca de exercícios de geometria *On-line*, de acesso livre, na qual poderiam ser catalogadas automaticamente diferentes *classes de soluções* para cada exercício. Do mesmo modo, poderiam ser catalogados os erros mais freqüentemente cometidos (uma biblioteca de erros comuns).

Referências Bibliográficas

Aleven, V., K. R. Koedinger, H. C. Sinclair, & J. Snyder (2004). Combatting shallow learning in a tutor for geometry problem solving. *1452*, 364–373.

Arcavi, A. & N. Hadas (2000). Computer mediated learning: an example of an approach. *International Journal of Computers ofr Mathematical Learning 5*(1), 25–45.

Barros, L. N. & E. T. Santos (2000). Um estudo sobre a modelagem do domínio de geometria descritiva para a construção de um sistema tutor inteligente. In *Anais do Simpósio Brasileiro de Informática na Educação*, pp. 259–268.

Bellemain, F. (2001). Geometria dinâmica: Diferentes implementações, papel da manipulação direta e usos na aprendizagem. In *Simpósio Nacional de Geometria Descritiva e Desenho Técnico & International Conference on Graphics Engineering for Arts an Design*, pp. 1314–1329.

Bellemain, F. (2002). O paradigma micromundo. In *Anais do Colóquio de História e Tecnologia no Ensino de Matemática*, pp. 51–62.

Berge, Z. L., L. Y. Muilenburg, & J. V. Haneghan (2002). Barriers to distance education and training: Survey results. *The Quarterly Review of Distance Education 3*(4), 409–419.

Botana, F. & J. L. Valcarce (2002). A dynamic-symbolic interface for geometric theorem discovery. *Computer & Education 38*(1–3), 21–35.

Brandão, L. O. (2002). Algoritmos e fractais com programas de geometria dinâmica. *Revista do Professor de Matemática 49*(2), 27–34.

Brandão, L. O. (2004). Programação geometrica: Uso da geometria dinâmica para programação. In *Anais do Colóquio de História e Tecnologia no Ensino de Matemática*, pp. 191–202.

Brandão, L. O. & S. Isotani (2003). Uma ferramenta para ensino de geometria dinâmica na internet: igeom. In *Anais do Workshop sobre Informática na Escola*, pp. 1476–1487. XXIII Congresso da Sociedade Brasileira de Computação.

Brandão, L. O. & S. Isotani (2004). Geometria interativa na internet através do igeom. In *Proceedings of the Webmedia & LA-Web Joint Conference*, pp. 291–292.

Brandão, L. O., S. Isotani, & J. G. Moura (2004a). Geometria dinâmica com o igeom: algoritmos geométricos, autoria e avaliação automática de exercícios. In *Mini-Cursos do Simpósio Brasileiro de Informática na Educação*. CDROM – 30 páginas.

Brandão, L. O., S. Isotani, & J. G. Moura (2004b). A plug-in based adaptive system: Saaw. *Lecture Notes in Computer Science 3220*, 791–793.

Brandão, L. O., S. Isotani, & J. G. Moura (2004c). Saaw: Adaptive learning system on the web. In *Proceedings of the Webmedia & LA-Web Joint Conference*, pp. 121.

Braviano, G. & M. H. W. L. Rodrigues (2002). Geometria dinâmica: uma nova geometria? *Revista do Professor de Matemática 2*(49), 22–26.

Cavanaugh, C. S. (2001). The effectiveness of interactive distance education technologies in k-12 learning: A meta-analysis. *International Journal of Educational Telecommunications 7*(1), 73–88.

Clements, D. H. (2000). From exercises and tasks to problems and projetcs - unique contributions of computers to innovative mathematics education. *Journal of Mathematical Behavior 19*(1), 9–47.

Crowley, M. L. (1987). The van hiele model of the development of geometric thought. In *Learning and Teaching Geometry, k-12*, pp. 1–16. National Council of Teachers of Mathematics.

Freire, P. (1987). *Pedagogia do Oprimido*, Volume 21. Rio de Janeiro: Paz e Terra.

Gao, X.-S. & C. Zhu (1998). Building dynamic mathematical models with geometry expert: Geometric transformations, functions and plane curves. In *Proceedings of Asian Technology Conference in Mathematics*, Disponível em http://www.atcminc.com/mPublications/EP/EPATCM98/ATCMP038/paper.pdf.

Gao, X.-S. & C. Zhu (1999). Building dynamic mathematical models with geometry expert - iii a geometry deductive database. In *Proceedings of Asian Technology Conference in Mathematics*, Disponível em http://www.atcminc.com/mPublications/EP/EPATCM99/ATCMP040/PAPER/paper.pdf.

Gelernter, H. (1963). Realization of a geometry theorem-proving machine. In *Computers and Thought*, pp. 134–152. New York: McGraw-Hill.

Gelernter, H., J. R. Hanson, & D. W. Loveland (1963). Empirical explorations of the geometry-theorem proving machine. In *Computers and Thought*, pp. 153–163. New York: McGraw-Hill.

Gibson, E. J., P. W. Brewer, A. Dholakia, M. A. Vouk, & D. L. Bitzer (1995). A comparative analysis of web-based testing and evaluation systems. In *Proceedings of Fourth International World Wide Web Conference*, http://renoir.csc.ncsu.edu/Faculty/Vouk/Papers/Gibson/Gibson_WWW4_1995.pdf.

Gilmore, P. C. (1970). An examination of the geometry theorem-proving machine. *Artificial Intelligence 1*, 171–187.

Gravina, M. A. (1996). Geometria dinamica - uma nova abordagem para o aprendizado da geometria. In *Anais do VII Simpósio Brasileiro de Informática na Educação*, pp. 1–13.

Grothman, R. (1999). *C.A.R - Compass And Rules*. http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/java/zirkel/.

Guerra, J. H. L. (2000). Utilização do computador no processo de ensino-aprendizagem: uma aplicação em planejamento e controle da produção. Dissertação de mestrado em engenharia de produção, Escola de Engenharia de São Carlos, Universidade de São Paulo.

Guimarães, L. C., R. Barbastefano, & E. Belfort (2002). Tools for synchronous distance teaching in geometry. In *Proceedings of Second International Conference on the Teaching of Mathematics*, http://www.math.uoc.gr/ ictm2/Proceedings/pap385.pdf.

Hannafin, R. D. (2001). Learning with dynamic geometry programs: Perspectives of teachers and learners. *The Journal of Educational Research 94*(3), 132–144.

Hara, N. & R. Kling (1999, 12). Student's frustrations with a web-based distance education course. *First Monday: Journal on the Internet 4*(12).

Hentea, M., M. J. Shea, & L. Pennington (2003). A perspective on fulfilling the expectations of distance education. In *Proceeding of the conference on Information technology curriculum*, pp. 160–167. ACM Press.

Hijazi, S. (2003). Interactive technology impact on quality distance education. *Eletronic Journal of e-Learning 1*(1), 35–44.

Holland, G. (2002). *geolog*. http://www.uni-giessen.de/ gcp3/geolog.htm.

Hollebrands, K. F. (2003). High school students' understandings of geometric transformations in the context of a technological environment. *Journal of Mathematical Behavior 22*(1), 55–72.

Isotani, S. & L. O. Brandão (2001). imática: Ambiente interativo de apoio ao ensino de matematica via internet. In *Anais do Workshop sobre Informática na Escola*, pp. 533–543. XXI Congresso da Sociedade Brasileira de Computação.

Isotani, S. & L. O. Brandão (2004a). Autoria e avaliação automática de exercícios no igeom. In *Proceedings of the Webmedia & LA-Web Joint Conference*, pp. 276–279.

Isotani, S. & L. O. Brandão (2004b). Ferramenta de avaliação automática no igeom. In *Anais do Simpósio Brasileiro de Informática na Educação*, pp. 328–337.

Isotani, S. & L. O. Brandão (2004c). igeom: Ferramenta de autoria e avaliação de exercícios. In *Anais do Simpósio Brasileiro de Informática na Educação*. CDROM – 1 página.

Isotani, S. & L. O. Brandão (2005). Analisando construções no igeom: Uma abordagem para correção automática de exercícios. In *Global Congress on Engineering and Technology Education*, pp. 1038–1042.

Jackiw, N. (1995). *The Geometer's Sketchpad v3.0*. Berkeley: Key Curriculum Press.

Java (2004). *Linguagem de Programação Java*. http://java.sun.com/: Sun Microsystems.

Jiang, Z. (1999). Explore geometry over the internet using javasketchpad. In *Proceedings of Asian Technology Conference in Mathematics*, Disponível em http://www.atcminc.com/mPublications/EP/EPATCM99/ATCMP003/PAPER/paper.pdf.

Jones, D. (1996). Computing by distance education: problems and solutions. *ACM SIGCSE Bulletin 28*(SI), 139–146.

Jones, K. (2000). Providing a foundation for deductive reasoning: students' interpretations when using dynamic geometry software and their evolving mathematical explanations. *Educational Studies in Mathematics 44*(1), 55–85.

Jones, K. (2003). Using the internet in the teaching and learning of mathematics: a research bibliography. *Micromath 19*(2), 43–44.

Kaczmarczyk, L. C. (2001). Accreditation and student assessment in distance education: why we all need to pay attention. In *ITiCSE '01: Proceedings of the annual conference on Innovation and technology in computer science education*, pp. 113–116. ACM Press.

Kappraff, J. (1991). *Connections: the geometric bridge betwwen art and sciense*. Mc Graw-Hill.

King, J. & D. Shattschneider (1997). *Geometry Turned On - Dynamic Software in Learning, Teaching and Research*. Washington: Mathematical Association of America.

Kirby, E. (1999). Student's frustrations with a web-based distance. In *Technology and Teacher Education Annual*, pp. 199–205.

Kortenkamp, U. (1999). *Foundation of Dynamic Geometry*. dissertation for ph.d. of technical sciences, Swiss Federal Institute of Technology Zurich, Disponível em http://kortenkamps.net/papers/diss.pdf.

Kortenkamp, U. (2001). The future of mathematical software. In *Proceedings of Multimedia Tools for Communicating Mathematics*, http://kortenkamps.net/papers/2001/future.pdf. Springer-Verlag.

Laborder, J. M. & F. Bellemain (1997). *Cabri Geometry II*. Dallas: Texas Instruments.

Lawhead, P. B., E. Alpert, C. G. Bland, L. Carswell, D. Cizmar, J. DeWeitt, M. Dumitru, E. R. Fahraeus, & K. Scott (1997). The web and distance learning: what is appropriate and what is not. *SIGCUE Outlook 25*(4), 27–37.

Lindquist, M. & A. Shulte (1987). *Learning and Teaching Geometry, k-12*. Reston: National Council of Teachers of Mathematics.

Litto, F. M. (2003). Educação a distância e a usp. *Jornal da USP 18*(639). Disponível em http://www.futuro.usp.br/producao_cientifica/artigos/fl_eadeausp.htm.

Litto, F. M., A. Filatro, & C. André (2004). Brazilian research on distance learning, 1999-2003: A state-of-the art study. In *Proceedings of International Congress of Distance Education*, http://www.abed.org.br/congresso2004/por/pdf/180-TC-D4.pdf.

Mandelbrot, B. (1983). *The fractal geometry of nature*. New York: W.H. Freeman.

Marrades, R. & A. Gutiérrez (2000). Proofs produced by secondary school students learning geometry in a dynamic computer environment. *Educational Studies in Mathematics 44*(1), 87–125.

Melo, L., J. M. Ferreira, & J. D. A. Pontes (2000). Um software educacional para o descobrimento de propriedades matemáticas. In *Anais do XX Congresso da Sociedade Brasileira de Computação*, pp. 1476–1487.

Moura, J. G. & L. O. Brandão (2004). Saw - sistema de aprendizagem pela web baseado em applets. In *Proceedings of the Webmedia & LA-Web Joint Conference*, pp. 280–283.

Munzner, T. (1996). Mathematical visualization: standing at the crossroads. In *Proceedings of the conference on Visualization*, pp. 451–453. IEEE Computer Society Press.

Nunes, I. B. (1994). Noções de educação à distância. *Revista Educação à Distância 32*(1), 7–25.

Oldknow, A. (1997). Dynamic geometry software - a powerful tool for teaching mathematics, not just geometry! In *Proceedings of International Conference on Technology in Mathematics Teaching*, http://euler.uni-koblenz.de/ictmt3/cd-rom/pdf/oldknow2.pdf.

Papert, S. (1999). *Mindstorms: children, computer and powerful ideas* (second edition ed.). New York: Basic Books.

Piesigilli, F., E. T. Santos, & L. O. Brandão (2002). Ferramentas para o ensino presencial e on-line de geometria descritiva. Trabalho de Formatura apresentado no Instituto de Matemática e Estatística da Universidade de São Paulo. Disponível em http://www.linux.ime.usp.br/ cef/mac499-02/monografias/fabi/.

Pólya, G. (1978). *Arte de resolver problemas : um novo aspecto do metodo matematico*. Rio de Janeiro: Editora Interciência.

Rodrigues, D. W. L. (2002). Uma avaliação comparativa de interfaces homem-computador em programas de geometria dinamica. Dissertação de mestrado em engenharia de produção, Universidade Federal de Santa Catarina.

Sahara, R. H. & L. O. Brandão (2001). Implementação de um programa de geometria dinâmica em java. Trabalho de Formatura apresentado no Instituto de Matemática e Estatística da Universidade de São Paulo. Disponível em http://www.linux.ime.usp.br/ cef/mac499-01/monografias/hideo/.

Santos, E. T., L. N. Barros, & V. C. P. N. Valente (2001). Projetando uma ontologia para geometria descritiva. In *Anais do Congresso Internacional de Engenharia Gráfica nas Artes e no Desenho*, pp. 918–128.

Santos, E. T. & J. I. R. Sola (2001). A proposal for an on-line library of descriptive geometry problems. *Journal for Geometry and Graphics 5*(1), 93–100.

Scapin, R. H. (1997). Desenvolvimento de uma ferramenta para criação e correção automática de provas na world-wide-web. Dissertação de mestrado em física, Instituto de Física de São Carlos, Universidade de São Paulo.

Sutcliffe, G. (2004). *Overview of Automated Theorem Proving*. http://www.cs.miami.edu/ tptp/OverviewOfATP.html.

Thomas, M. D., P. R. Patel, A. D. Hudson, & D. A. B. Jr. (1996). *Java Programming for the Internet*. Durham: Ventana Communications Group.

Usiskin, Z. (1987). Resolving the continuing dilemmas in school geometry. In *Learning and Teaching Geometry, k-12*, pp. 17–31. National Council of Teachers of Mathematics.

Valente, V. C. P. N. (2003). *Desenvolvimento de um ambiente computacional interativo e adaptativo para apoiar o aprendizado de geometria descritiva*. Tese de doutorado em engenharia, Escola Politécnica da Universidade de São Paulo.

Ávila, G. (1999). Os paradoxos de zenão. *Revista do Professor de Matemática 2*(39), 9–16.

Zirkle, C. (2004). Access barriers experienced by adults in distance education courses and programs: A review of the research literature. In *Proceedings of idwest Research-to Practice Conference*, http://www.iupui.edu/ adulted/mwr2p/program/Proceedings/Zirkle.pdf.

Índice Remissivo