# Esboço de Fluxogramas no Ensino de Algoritmos

**Halley Wesley A. S. Gondim, Ana Paula Ambrósio**

Instituto de Informática – Universidade Federal de Goiás (UFG)
Campus II – Samambaia – Caixa Postal 131 – CEP 74001-970 Goiânia – GO - Brasil
halley_was@yahoo.com.br, apaula@inf.ufg.br

***Abstract.*** *Algorithms courses have one of the greatest reproval and drop out rates in Computer Science programs. The lack of motivation is one of the factors, due mainly to the difficulties students face in the development of the necessary logic/algorithmic reasoning. This paper presents a teaching tool that targets this problem by using digital ink for the creation of fluxograms that are converted to programming code that can in turn be tested and analyzed.*

***Resumo.*** *A disciplina de Algoritmos possui um dos maiores índices de reprovação em computação. A falta de motivação é um dos grandes fatores, já que os alunos enfrentam dificuldades no desenvolvimento do raciocínio lógico/algorítmico necessário. Este artigo propõe uma ferramenta de ensino que ameniza tais problemas com a utilização de tinta digital para a criação de esboços de fluxogramas que são convertidos para códigos podendo ser testados e avaliados.*

## 1. Introdução

A disciplina Algoritmos e Programação de Computadores sempre representou um desfio para os cursos de Computação. De fato, este é considerado um dos sete grandes desafios do ensino da computação [McGettrick et al., 2005]. Ensinar programação básica no ensino superior tem servido de tema para muitas discussões entre professores da disciplina [Astrachan et al. 2005], [Bailie et al, 2003], [Bruce, 2005] e [SIGCSE-members, 2005]. Já resultou, inclusive, em recomendações curriculares feitas pela ACM (American Association for the Computing Machinery) e pelo IEEE (Institute of Electrical and Electronics Engineers, Inc).

Problemas com esta disciplina são apontados como responsáveis pelo grande número de reprovações e desistências em cursos de Computação [Kumar, 2003]. Isto se torna mais preocupante quando se verifica que o número de alunos matriculados em cursos de Computação tem decrescido a nível mundial [Denning, 2004]. Trabalhar esta disciplina torna-se portanto vital para o sucesso dos cursos na área.

Atualmente, quando alunos ingressantes nos cursos de Computação se deparam com uma disciplina que envolve raciocínio lógico e/ou algorítmico, sintaxe pesada e tecnologia, eles acabam encontrando grandes dificuldades na aprendizagem [Rocha 1991] e [Chen e Morris 2005]. A metodologia atual de ensino é geralmente pouco estimulante, e trabalha com resolução de problemas pouco atrativos [Chen e Morris 2005]. Além disto, por ser uma disciplina que envolve muito o professor, que precisa

acompanhar de perto o trabalho dos alunos, e muitas vezes não consegue acompanhar toda a turma, alguns alunos acabam ficando para trás. Isto gera desânimo, evasão e reprovação [Gomes 2000].

Por outro lado, sabe-se que o ser humano tem mais habilidade para interpretar um algoritmo usando formas visuais que em formato de código texto. Isto se deve ao fato que o hemisfério esquerdo do cérebro processa informação oral e lógica, enquanto o hemisfério direito processa informação visual e espacial. Quando o aluno interpreta o código texto, o hemisfério direito não contribui significativamente no processo de aprendizagem. [Da Silva 2001]. Assim, a chave para fazer o ensino de algoritmo mais acessível está em utilizar aspectos visuais e textuais, estimulando ambos os hemisférios do cérebro.

Neste sentido, a ferramenta proposta integra fluxogramas e tinta digital. O uso de fluxogramas se deve a três razões principais. Primeiro: fluxogramas possuem uma sintaxe mínima. Quando se reduz o foco em sintaxe, pode-se aumentar o esforço em análise. Segundo: fluxogramas é uma representação universal. Nenhum outro sistema visual alcançou a aceitação dos fluxogramas. Terceiro: fluxogramas são mais fáceis para estudantes iniciantes em computação do que estrutura de código.

Estudo realizado por [Crews e Ziegler 1998] verificou que estudantes iniciantes em algoritmos cometem menos erros, têm maior confiança e resolvem problemas representados por algoritmos simples e médios mais rapidamente, quando utilizam fluxogramas. Já em [Scanlan 1989] verificou-se que o uso de fluxogramas leva a uma diminuição de erros em algoritmos complexos.

Apesar de terem sido muito usados no processo de programação até os anos 80, hoje os fluxogramas não são mais utilizados no ensino de algoritmos. Segundo [Scanlan 1989] o declínio dos fluxogramas como representação de algoritmos foi provocado principalmente pelo trabalho de [Shneiderman et al 1977] que em seu estudo concluiu que fluxogramas não transmitem nem mais nem menos informações que código de programação, e, portanto, não era útil para programadores. No entanto Scanlan contesta estes resultados. Ao realizar sua experiência com 554 alunos, encontrou: tempo menor para a compreensão, menor número de erros, ganho na confiança e queda no tempo para a resolução de algoritmos Médios e Complexos. Segundo Crews/Ziegler e Scanlan , os alunos preferiram a utilização de fluxogramas ao invés de código. [Medina e Fertig 2005] concordam com estes resultados, verificando que fluxogramas são bastante úteis para representações de algoritmos em um nível alto de abstração.

Analisando os estudos de [Da Silva 2001], [Crews e Ziegler 1998] e [Scanlan 1989], conclui-se que a melhor forma de se assimilar um conteúdo é por meio de visualizações gráficas. Neste contexto, o aluno pode fazer anotações em forma visual (desenhos, setas, observações, comentários) de tudo aquilo que acredita ser importante, para que possa ser facilmente relembrado posteriormente. Algumas ferramentas que exploram estas potencialidades podem ser encontradas em [Cares 2002] e [Mendes-Gomes 2002].

Apesar do fluxograma ser uma das formas mais simples de se observar, criar e apresentar um algoritmo em forma gráfica, o uso de ferramentas computacionais tradicionais reduz a naturalidade e a liberdade do usuário na expressão de suas idéias se

comparado ao uso de lápis e papel. No entanto, o surgimento de novas tecnologias que utilizam tinta digital pode reproduzir esta realidade, auxiliando os estudantes a visualizarem elementos, argumentos, criação de consistência e claridade que não existem sem a ajuda de objetos visuais, com a vantagem de criar um registro permanente do trabalho sendo efetuado.

Neste sentido, este trabalho visa a utilização de tinta digital para a criação de esboços de fluxogramas tornando as anotações mais prazerosas e úteis. Este esboço é convertido por meio de técnicas de reconhecimento de formas, que são então interpretadas, eliminando os longos testes feitos sobre o papel, tornando o trabalho mais interessante e prático [Medeiros e Dazzi 2000]. Os fluxogramas podem também serem convertidos em código de máquina específico, criando uma assimilação entre como se desenvolve uma solução de um algoritmo por meio de um fluxograma e como se desenvolve a solução por meio de um código ou pseudocódigo [Santiago 2004][Chen e Morris 2005].

Este artigo está dividido em quatro seções. Na segunda seção é apresentado o reconhecimento de um esboço por meio de técnicas que identificam desenhos feitos com tinta digital. Na terceira seção é feita uma descrição da ferramenta proposta, observando a sua grande contribuição para o aprendizado, somando as vantagens da tinta digital com o rico conteúdo de informações visuais dos fluxogramas.

## 2. Reconhecimento

Esboços são figuras informais criadas a mão com o intuito de resolver um determinado problema. Um esboço pode ser feito de diversas maneiras, e a forma do esboço pode variar de pessoa para pessoa. Isto é, assim como na escrita, cada pessoa tem características próprias que se refletem no esboço. Por exemplo, a Figura 1 apresenta o esboço de um retângulo feito de quatro modos diferentes. Além disso, por trabalhar com a escrita manual, o sistema está sujeito a ruídos. Nestes casos os ruídos surgem da inabilidade da pessoa de desenhar com precisão linhas e curvas.

Por isto, o reconhecimento de esboço não é fácil e intuitivo. Para amenizar estes problemas algumas soluções podem ser adotadas. Uma solução é não atribuir muita liberdade no esboço, ou melhor, quanto menor o número de restrições a uma forma mais será difícil sua interpretação. Outra solução é não exigir o término de um objeto antes de se começar o próximo. Assim a pessoa pode, ao longo do esboço, acrescentar mais traços que auxiliam na conclusão. Como exemplo: ao se desenhar um carro, pode-se desenhar uma roda, depois o chassi, janelas e por fim a outra roda. Neste caso o interpretador reconhece a forma carro e não somente círculos, polígonos dentro outras. [Davis 2007].

O sistema de reconhecimento de traços *Natural Interaction* (ou Magic Paper) [Davis 2007] utiliza uma escala de mão que é na verdade uma linguagem de restrições de formas que descreve o esboço, observando a direção, curvatura e a velocidade do esboço (Figura 2). A linguagem de reconhecimento de esboço permite descrever formas mais complexas a partir de formas primitivas, associadas a um contexto. Podemos entender seu funcionamento na Figura 3a. Por exemplo, a seta é a união de três linhas conectadas e com tamanhos específicos (*shalft, head 1 e 2*), Neste caso, as formas primitivas são as linhas e por saber que neste determinado contexto setas servem para

conectar conceitos, ao encontrar três linhas que se organizam de forma similar a uma seta, deduz-se que seja uma seta.

Em [Paulson et al 2007] são propostos o reconhecimento de oito formas primitivas no sistema LADDER: linhas, poli-linhas, círculo, elipse, arco, curva, mola e espiral. Para estas formas, os autores alcançaram um grau de satisfatibilidade de até 98,8% no reconhecimento das mesmas utilizando o sistema desenvolvido por eles.

Com base nesse trabalho é possível criar um reconhecedor para domínios específicos. Na Figura 4 podemos encontrar um reconhecedor de modelos UML. Por enquanto o reconhecimento de esboço de maneira geral não é possível, pois a compreensão dos mesmos não é satisfatória. Para ocorrer a identificação de um esboço de maneira geral, é necessário testar de forma exaustiva o significado de cada traço, ocasionando a inviabilidade deste tipo de reconhecimento. No caso de um reconhecedor de formas de fluxograma, a definição do contexto é viável, pois possui um número limitado de formas, eliminando a exaustão de testes.

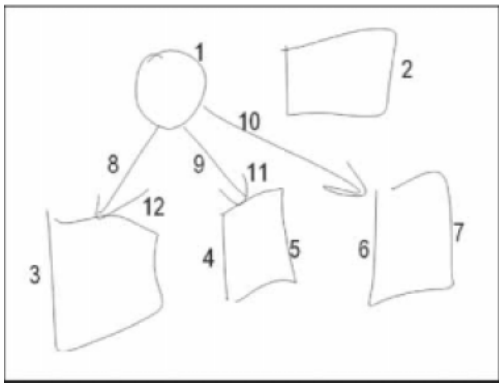


**Figura 1. Diferentes formas de esboço de um retângulo**

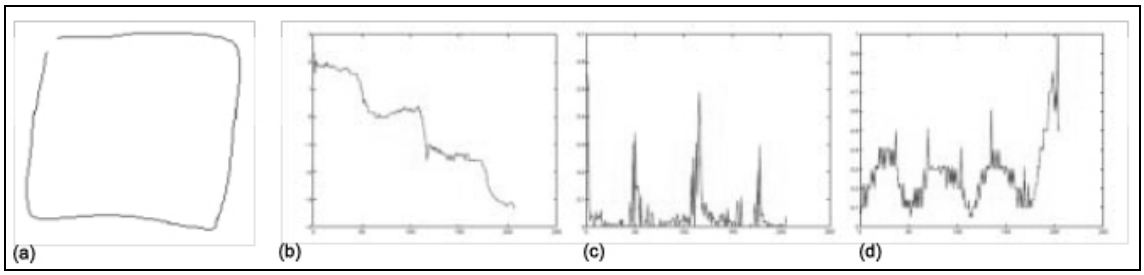


**Figura 2. Características de um esboço. (a) esboço de um quadrado e a direção do desenho (b), (c) curvatura e (d) velocidade [Davis 2007].**

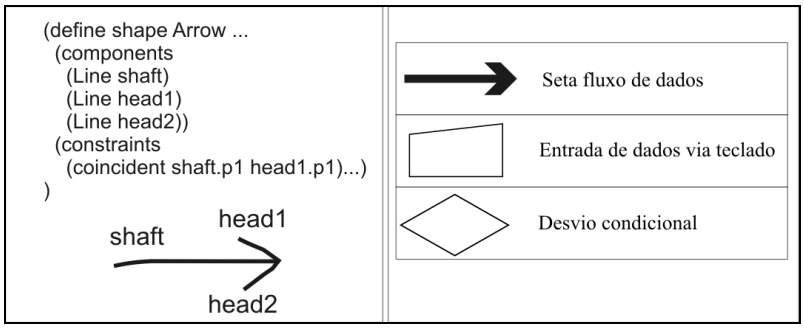


**Figura 3. A) Exemplo de uma linguagem para reconhecimento de uma seta [Paulson 2007]. B) Exemplo de algumas formas básicas de um fluxograma**

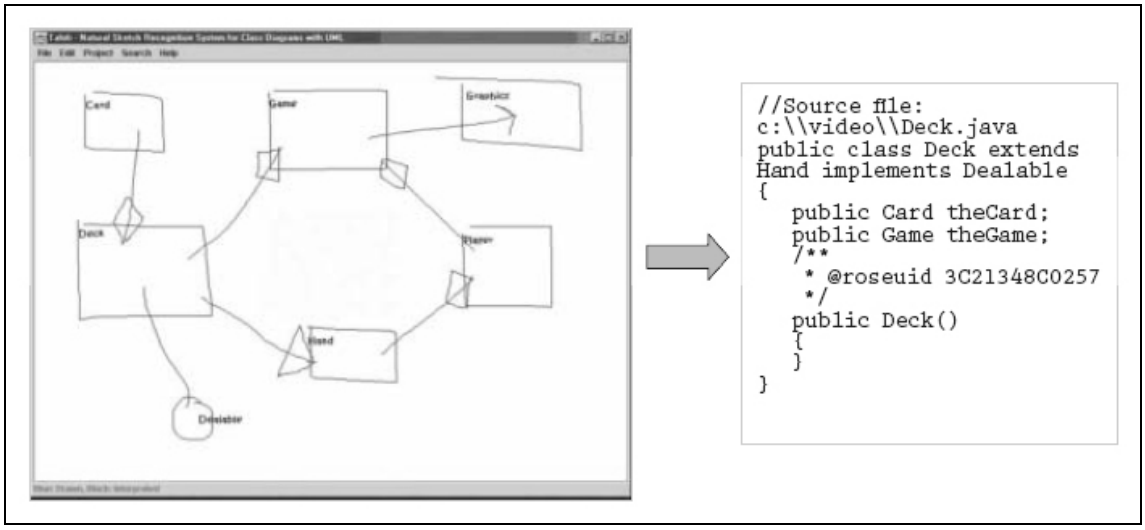


**Figura 4. Esboço de UML com seu respectivo código após o reconhecimento utilizando RationalRose. [Davis 2007]**

## 3. A Ferramenta

Com a introdução do uso de tablets e outros dispositivos que possuem tinta digital em salas de aula, abre-se a possibilidade de re-introduzir a escrita na solução de problemas. A tinta digital deixa que os estudantes se expressem de vários modos. Os efeitos desta liberdade são evidentes ao longo das atividades em sala (Figura 5). Sua flexibilidade de expressão permite aos estudantes incluir diagramas, símbolos matemáticos, comentários ou desenhos complexos em suas respostas. Um dos impactos importantes da tinta digital no ambiente de sala de aula é a possibilidade de se identificar o trabalho de um estudante de modo exclusivo. [Anderson et al 2007].

O ambiente proposto visa a confecção de esboços de fluxogramas utilizando a tinta digital. Mas, se esboços são tão informais e pessoais, então qual o motivo de se criar uma ferramenta de esboço ao invés de utilizar as atuais ferramentas que criam fluxogramas de modo mais preciso? Pode-se citar dois motivos. O primeiro é que a utilização de uma caneta sempre será uma forma mais natural do que um teclado ou mouse, deixando de forma livre e mais intuitiva as imagens geradas, aliando a facilidade

de pensamento à facilidade de esboçar. Segundo e mais importante, os sistemas que possuem as formas de um fluxograma possuem limitações impostas pelos projetistas [Davis 2007]. Por exemplo, é desnecessário se preocupar com tamanho e ângulos precisos, sendo que o objetivo é somente esboçar uma idéia.

O ambiente também possui um reconhecedor das formas básicas do fluxograma e um conversor para uma linguagem. O reconhecedor permite transformar um esboço em um fluxograma gráfico, eliminado as ambigüidades que um desenho possa gerar dada sua informalidade. Este fluxograma não ambíguo pode então ser convertido para uma linguagem de programação permitindo que seja testado, verificando se está correto ou não. Isto permite que iniciantes em algoritmos possam testar a lógica de solução do problema (algoritmo) de modo intuitivo e fácil.

Como cada forma gráfica básica reconhecida do fluxograma possui semântica específica, que se traduz em uma estrutura das linguagens de programação, na execução de um fluxograma, o interpretador analisa o código montado em memória e o executa [Santiago 2004][Chen e Morris 2005]. Assim, o interpretador nada mais é que a definição de uma linguagem possuindo um conjunto de instruções.

Como o fluxograma possui uma gramática idêntica à da interpretação, é possível realizar a conversão para uma pseudo-linguagem ou para uma linguagem de máquina específica, por exemplo, Pascal [Santiago 2004]. Desta forma o aluno ao esboçar o algoritmo utilizando o auxílio da tinta digital pode, além de ter mais facilidade na anotação, escolher converter o fluxograma para uma linguagem específica, trazendo assim um facilitador para o primeiro contato com a construção de códigos.

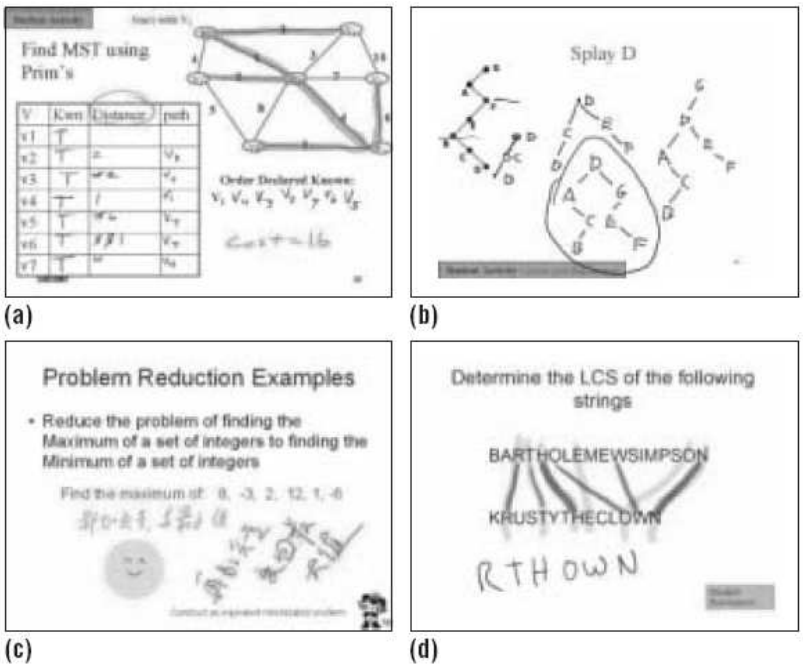


**Figura 5. Modelos de esboços com a utilização de tinta digital, deixando de forma natural a produção de anotações conforme o raciocino e a linguagem do estudante. [Anderson et al 2007]**

O processo é dividido em dois módulos. No primeiro o estudante esboça o fluxograma de forma livre com o auxílio da caneta. No segundo é feita a interpretação do esboço e também a disponibilização em formato código. No primeiro módulo o estudante cria de forma natural o fluxograma, contendo as funções básicas de desenho como apagar, recortar, colar, mudar cor do traço e selecionar a área para interpretação (execução de trechos específicos). Somente após o término de todo o esboço o estudante seleciona o botão rodar, e o mesmo será interpretado. Caso o reconhecimento não seja satisfatório o programa oferece uma lista de sugestões para o esboço que não foi reconhecido, assim o aluno pode escolher o que realmente representa o esboço.

Durante a interpretação, se existir erro sintático ou léxico no esboço o mesmo será avisado em qual parte do fluxograma se encontra o erro. O estudante pode então selecionar o modo esboço e realizar a correção. Não ocorrendo nenhum erro na interpretação do fluxograma o estudante pode modificar as cores dos componentes do fluxograma, realizar testes de mesa, verificando se realmente o algoritmo atende as suas necessidades. Caso o estudante queira compactar um determinado trecho do fluxograma o mesmo somente precisa selecionar os elementos e escolher compactar, desta forma ocorre um ganho no espaço. Quando houver a necessidade de visualizar os objetos compactados é necessário somente executar o comando inverso. Logo após o término do teste o aluno pode converter o fluxograma em um código específico pressionando o botão converter para código. Um exemplo desta aplicação está na Figura 6 que apresenta a criação de um esboço, o reconhecimento do desenho e a conversão para um código ou pseudo-código.

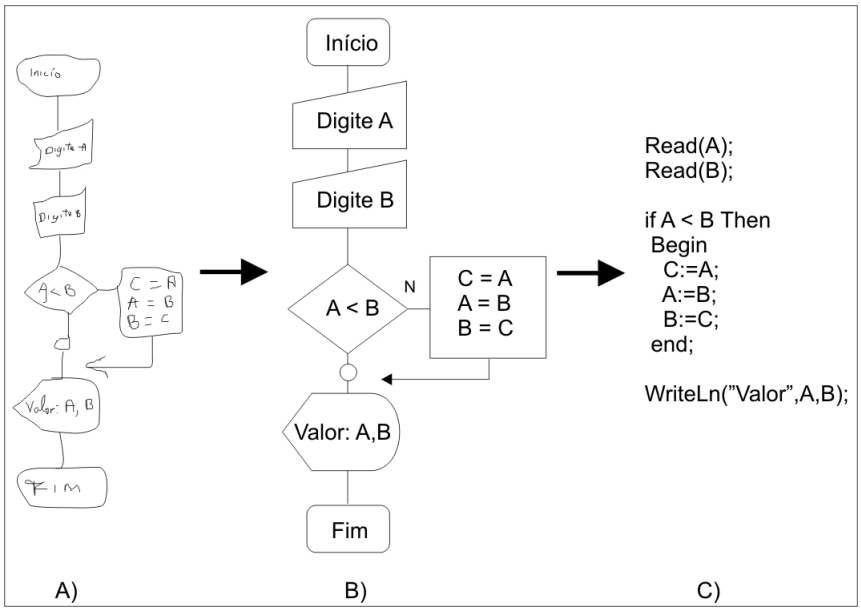


**Figura 6. Exemplo de esboço de fluxograma (A), reconhecimento do desenho (B) e conversão para código (C).**

## 4. Conclusão

O trabalho proposto apresenta uma ferramenta para ser utilizada em cursos iniciais de Algoritmos e Programação de Computadores. Nela o aluno pode esboçar fluxogramas utilizando tinta digital que serão convertidos para código de programas que podem ser testados, validando o raciocínio algorítmico do aluno.

Como já foi verificado em estudos anteriores existe uma grande melhoria na aprendizagem da disciplina de Algoritmos com a introdução de novas formas de trabalho, tanto por meio de da inclusão de fluxogramas [Santiago 2004][Davis 2007] [Crews e Ziegler 1998][Chen e Morris 2005][Scanlan 1989], quanto de tablets ou dispositivos que possuam tinta digital para ministrar aulas. [Turner 2006][Anderson 2001].

É importante ressaltar que, ao se motivar o aluno com a utilização de tinta digital no início de seu aprendizado, ele ganhará afeição ao estudo, assim como introduzir ferramentas que o auxiliam no desenvolvimento de problemas causará um impacto positivo na forma de resolver o problema, tornando o aprendizado mais divertido e proveitoso [Rapkiewicz e Turner 2006].

A ferramenta encontra-se em fase de implementação e deverá ser testada nas disciplinas de Algoritmos e Programação de Computadores da Universidade Federal de Goiás.

## Referências

Anderson R, et al; (2007) Classroom Presenter: Enhancing Interactive Education with Digital Ink. University of Washington.

Astrachan, O., Bruce, K., Koffman, E., Kölling, M., & Reges, S. (2005). Resolved: Objects early has failed. SIGCSE '05: Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, St. Louis, Missouri. 451-452.

Bailie, F., Courtney, M., Murray, K., Schiaffino, R., & Tuohy, S. (2003). Objects first - does it work? Journal of Computing in Small Colleges, 19(2), 303-305.

Bruce, K. B. (2005). Controversy on how to teach CS 1: A discussion on the SIGCSE-members mailing list. SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education), 37(2), 111-117.

Cares, P. L. L. (2002) Ambiente para teste de mesa utilizando fluxograma. Trabalho de Conclusão Graduação)–Faculdade de Ciência da Computação, Universidade do Vale do Itajaí.

Chen, S. e Morris, S. (2005). Iconic programming for flowcharts, java, turing, etc. In Proceedings of the 10th Annual SIGCSE Conference on innovation and Technology in Computer Science Education. ACM Press, 104-107.

Crews T. e Ziegler U. (1998) The Flowchart Interpreter for Introductory Programming Courses. Department of Computer Science Western Kentuchy University Bowling Green.

Da Silva, M. A. V.; et al; (2001) MODI – A Proposal of a visual tool to simulate and synthesize software applied to embedded systems. DEMIC / FEEC / UNICAMP.

Davis R; (2007) Magic Paper: Sketch-Understanding Research. Massachusetts Institute of Technology.

Denning, P. J. (2004). The field of programmers myth. Communications of the ACM, 47(7), 15-20.

Gomes, A. J. (2000) Ambiente de Suporte à aprendizagem de Conceitos Básicos de Programação, dissertação (Mestrado). Universidade de Coimbra.

Kumar, A. N. (2003). The effect of closed labs in computer science I: An assessment. Journal of Computing in Small Colleges, 18(5), 40-48.

McGettrick, A., Boyle, R., Ibbett, R., Lloyd, J., Lovegrove, G., & Mander, K. (2005). Grand challenges in computing: Education--A summary. The Computer Journal, 48(1), 42-48.

Medeiros, C. L.; Dazzi, R. L. S. (2002) Aprendendo algoritmos com auxílio da WEB, em: Congresso Brasileiro de Computação 2002, Itajaí. Anais... Itajaí: UNIVALI – CTTMar.

Medina M. e Fertig C. (2005) Algoritmos e Programação: Teoria e Prática São Paulo: Novatec, pág. 21.

Mendes, A. J. N.; Gomes, A. J. (2000) Suporte a aprendizagem de programação com o ambiente SICAS. Em: Congresso Ibero Americano de Informática Ducativaribie, 5., 2000, Viña del Mar-Chile. Anais... Viña del Mar-Chile: Universidad del Chile.

Paulson, Brandon E Hammond, Tracy. (2007) A System for Recognizing and Beautifying Low-level Sketch Shapes Using NDDE and DCR.

Paulson, Brandon E Hammond, Tracy. (2007) Marqs: Retrieving Sketches Using Domain- And Style-Independent Features Learned From A Single Example Using A Dual-Classifier.

Rapkiewicz, Clevi Elena; et al. (2006) Estratégias Pedagógicas no Ensino de Algoritmos e Programação Associadas ao uso de Jogos Educacionais.

Rocha, H. V. (1991) Representações Computacionais Auxiliares ao Entendimento de Conceitos de Programação. UNICAMP.

Santiago, Rafael e Dazzi, Rudimar L.S. (2004) Ferramenta de Apoio ao Ensino de Algoritmos.

Scanlan, A. D. (1989) Structured Flowcharts Outperform Pseudocode: An Experimental Comparisons. California States University at Sacramento.

Shneiderman B. et al, (1977) "Experimental Investigations of the Utility of Detailed Flowcharts in Programming." Comm. ACM, pp. 373-381

SIGCSE-members. (2005). Archives of sigcse-members@ACM.ORG. Retrieved March 22, 2006, from http://listserv.acm.org/archives/sigcse-members.html

Turner S, et al; (2006) Note Taking and the Tablet PC. Virginia Tech – Computer Science Department.