RECONHECIMENTO DE PADRÕES EM APLICAÇÕES DE REALIDADE AUMENTADA USANDO REDE NEURAL ARTIFICIAL

CELSO CAMILO°, WNÊITON L. GOMES*, ALEXANDRE CARDOSO*, EDGARD L. JR*, KEIJI YAMANAKA*

**Faculdade de Engenharia Elétrica - Universidade Federal de Uberlândia*
Av. João Naves de Ávila, 2121 – Campus Santa Mônica – Universidade Federal de Uberlândia – MG - Brasil
Cep 38400-902 – Fone(34) 32394714

°Faculdade de Ciências Exatas e Tecnologia - Universidade Federal da Grande Dourados
Fone(67) 3411-3901- Dourados, MS – Brazil.

E-mails: celsocamilo@gmail.com, wneiton@yahoo.com.br,
alexandre,lamounier,keiji@ufu.br

Abstract - Several RNAs can be applied to many problems and areas, among them the patterns recognition (PR). The PR objective is the classification of objects (patterns) in a number of categories or classes. The patterns, in the applications of Augmented Reality (AR), are used to identify the point of virtual objects interference in the real world, thus, the system should recognize the patterns inside of the captured image, to make possible the interaction between the virtual and real worlds. Therefore, the objective of this work is to present the ANN as alternative for the patterns recognition in a AR system. The complexity analysis demonstrated a smaller execution time, benefit that is of fundamental importance during the execution of real time AR applications.

Keywords - Pattern Recognition, Artificial Neural Network, Augmented Reality.

Resumo – Muitos são os problemas e áreas que se pode aplicar as mais diversas RNAs, entre eles o reconhecimento de padrões (RP). O RP é a área de pesquisa que tem por objetivo a classificação de objetos (padrões) em um número de categorias ou classes. Nas aplicações de realidade aumentada (RA) é comum o uso de padrões para identificar o ponto de interferência dos objetos virtuais no mundo real, sendo assim, o sistema deve reconhecer os padrões dentro da imagem capturada, para possibilitar a interação entre os mundos virtual e real. Assim, o objetivo deste trabalho é apresentar a implementação de uma Rede Neural Artificial como alternativa para o reconhecimento de padrões em um sistema de RA. A análise de complexidade demonstrou um menor tempo de execução, benefício que é de fundamental importância durante a execução de aplicações de RA em tempo real.

Palavras-chave - Reconhecimento de padrões, Redes Neurais Artificiais, Realidade Aumentada.

1 Introdução

Atualmente a computação tem contribuído fortemente para resolução de problemas em diversas áreas do conhecimento.

Dentre as várias técnicas da computação, as Redes Neurais Artificiais (RNA) são baseadas em uma família de modelos computacionais inspirados em neurônios biológicos de seres humanos, no qual tem como principal característica o aprendizado (Fausett, 1994).

São diversos os problemas para aplicações de RNA, entre eles, a classificação de padrões e otimização (Fausett, 1994).

Em particular, na área da Computação Gráfica, os modelos de mundos virtuais utilizando técnicas de Realidade Virtual (RV) e Realidade Aumentada (RA) têm recebido merecido destaque devido a suas aplicações em psicologia, educação, processos industriais, cinema, entretenimento, entre outros (Azuma, 1997; Kirner, 1999).

A Realidade Aumentada é definida como uma combinação da visão do ambiente real com o ambiente virtual (Azuma, 1993; Boman, 1995; Feiner et al., 1993).

Em realidade aumentada a imagem de vídeo é processada e o ambiente real é "aumentado" com cenas gráficas geradas por computador (Bajura e Neumann, 1995).

A maioria dos projetos de baixo custo em RA são desenvolvidos utilizando um conjunto de bibliotecas chamado ARToolKit (Kato e Billinghurst, 2000).

Baseado em técnicas de reconhecimento de padrões adotado pelo ARToolKit e a capacidade que as RNA oferecem, o trabalho objetivo-se na busca por melhorias no desempenho do módulo de reconhecimento do ARToolKit. Uma vez que a captura da cena e o incremento de objetos virtuais são feitos em tempo real, há uma necessidade constante de redução do custo computacional. Por isso, foi feito a implementação de uma biblioteca utilizando técnicas de RNA com intuito de criar uma alternativa e obter melhor desempenho para reconhecimento de marcadores, comparado ao algoritmo tradicional do ARToolKit.

O trabalho está organizado da seguinte forma: Na seção 2 serão descritos conceitos básicos do ARToolKit e

como se dá o reconhecimento dos marcadores. A seção 3 introduz conceitos de redes neurais artificiais seguindo com detalhes do desenvolvimento do projeto e a análise de complexidade dos algoritmos RA e RNA.

2 Descrição da Biblioteca Artoolkit - Augmented Reality Toolkit

O ARToolKit é um conjunto de bibliotecas em linguagem C desenvolvidas na Universidade de Washington, sendo projetado para o desenvolvimento de aplicações de RA.

Através do uso de técnicas de visão computacional o ARToolKit calcula a posição e a orientação de marcadores ou padrões impressos em cartões, sendo esta posição e orientação capturadas por uma câmera digital de forma que estes marcadores venham a ser corretamente revestidos por objetos virtuais tridimensionais. A figura 1 mostra como é o processo de execução do ARToolKit.

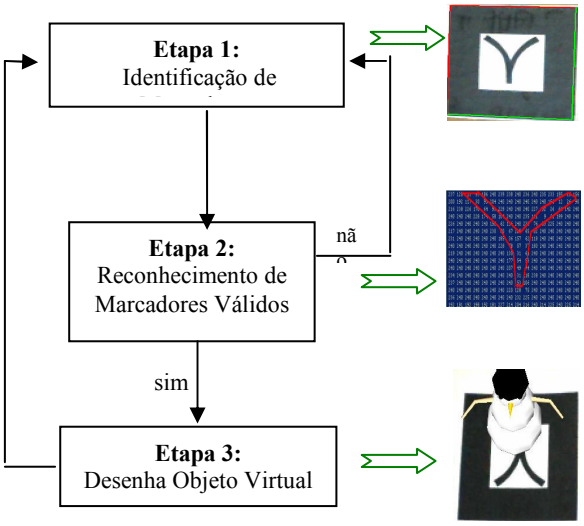


Figura 1. Fluxo de funcionamento do ARToolKit.

A primeira etapa da Figura 1 mostra à identificação dos marcadores, nesta fase a imagem capturada pela camera é toda investigada com o intuito de encontrar bordas que formem um quadrado. Todo quadrado encontrado é identificado como um marcador, no entanto, nem todos os marcadores (pré-cadastrados), são válidos, ou seja, podem existir marcadores que não foram cadastrados e por isso não devem representar nenhum objeto virtual.

Para ser considerado um marcador válido a figura contornada pelo quadrado identificado tem que ser igual ao pré-cadastrado (Etapa 2). Caso seja um marcador válido, na etapa 3 o objeto virtual relacionado com o marcador é desenhado. Caso não haja nenhum marcador válido, o processo de captura de marcadores (Etapa 1) é retomado.

Na seção 2.1 será descrito com mais detalhes o processo de reconhecimento do ARToolKit.

2.1 Reconhecimento de marcadores válidos com ARToolKit

A presente seção apresenta o processo de reconhecimento dos marcadores válidos pelo ARToolKit, que vai da identificação da figura, contornada pelas bordas, até o preenchimento de sinalizadores que associem o marcador válido ao objeto virtual a ser desenhado.

Todos os passos para determinar um marcador como válido, pode ser observado na Figura 2 a seguir.

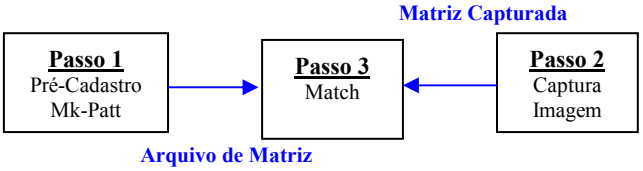
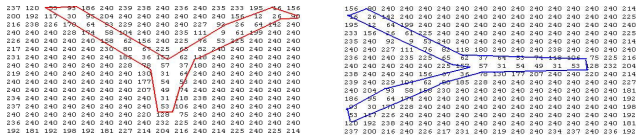


Figura 2. Processo de reconhecimento de marcadores.

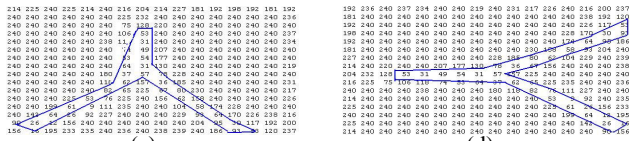
O passo 1 é uma fase de pré-execução, onde é feito o cadastro dos marcadores que serão usados na execução do sistema. Este pré-cadastro é feito por intermédio de um módulo do ARToolkit chamado Mk-Patt, que gera um arquivo de matrizes. São 12 matrizes de dimensões 16 x 16. Cada matriz determina uma posição do marcador, seja ela translação ou rotação. Com isso, não importa a forma como o marcador é capturado pela câmera, porque muitas das possibilidades de posicionamento do marcador estão previamente definidas nesse arquivo.

As matrizes são preenchidas de números que variam de 0 até 255 numa escala de cinza. Baseado em um limiar, é definido qual pixel na imagem é claro ou escuro, formando assim, um arquivo binário. Por exemplo, se o limiar for igual a 100, todo campo da matriz que tiver valor maior ou igual a 100 é convertido para 1 (branco) e todo campo que tiver valor menor a 100 recebe 0 (preto).

O arquivo de matrizes é organizado da seguinte forma: são quatro grupos de três matrizes de 16 x 16, totalizando doze matrizes para cada marcador cadastrado. Cada grupo tem três matrizes, sendo cada uma em uma posição de translação e a cada mudança de grupo tem uma rotação no marcador. Como exemplo, será apresentado, na figura 3, uma matriz de cada grupo. Considerando o limiar igual a 100, a figura do marcador foi realçada pelo contorno para que seja possível a observação da movimentação do desenho pelas matrizes.



(a) (b)



(c) (d)

Figura 3 (a,b,c,d). Arquivo de Matrizes do marcador Kanji.

Para simplificar a Figura 3, mostra apenas uma matriz de cada grupo. Como pode ser visto na Figura 3 (a,b,c e d), no passo um (Figura 2) é aplicado rotação nos desenhos da matriz a cada mudança de grupo. Como o marcador é quadrado e tem quatro vértices, portanto, são quatro rotações.

Desse modo são geradas 12 matrizes em diferentes posições de translação e rotação para cada marcador, possibilitando ao usuário observar objetos virtuais em diferentes posições.

No passo 2 (Figura 2) é gerado também uma matriz de dimensões 16x16 da parte interna do quadrado, no entanto, referente ao marcador capturado pelo dispositivo de câmera em tempo de execução. Essa matriz é enviada para a função Match (passo 3).

A função Match é o grande responsável pela identificação do marcador válido, além de definir a probabilidade de que este marcador seja o pré-cadastrado. A probabilidade é essencial, no caso de apresentação de dois marcadores similares, para definir em qual marcador será desenhado o objeto virtual. O marcador com maior probabilidade é o escolhido.

A identificação, feita pela função Match (passo 3), consiste basicamente de uma comparação entre o arquivo de matrizes de marcadores pré-cadastrados (passo 1) e a matriz capturada pelo dispositivo de câmera (tempo de execução) no passo 2. Se houver alguma semelhança, então o ARToolKit considera que encontrou um dos marcadores de referência (pré-cadastrados).

3 Redes Neurais Artificiais

As RNA's são técnicas computacionais que apresentam um modelo inspirado na estrutura neural de seres inteligentes e que adquirem conhecimento através da experiência, assim como um ser humano.

As RNA's têm sido utilizadas em varias áreas do conhecimento, como exemplo, a indústria, negócios, finanças, medicina, e dentre essas áreas os problemas mais comuns são a classificação, predição, reconhecimento de padrões e controle.

Assim como os neurônios do cérebro humano, as RNA's também possuem uma estrutura de neurônios, no entanto, matemáticos e artificiais. Esses neurônios ligados por conexões sinápticas são divididos em neurônios de entrada, neurônios internos e neurônios de saída, pelos quais a rede recebe sinal do meio externo, processam as informações e comunica com o exterior.

As arquiteturas mais típicas são Mono-camada, com neurônios de entrada e saída, e multi-camada, com neurônios de entrada, internos e de saída.

3.1 Inserindo a Rede Neural Artificial no reconhecimento dos marcadores válidos

Como foi visto no item 2, o ARToolKit tem uma etapa (etapa 2) de reconhecimento de padrões válidos, que foi descrito no item 2.1. No entanto, este trabalho propõe a substituição do algoritmo usado, para reconhecimento, por uma RNA (Figura 4), propiciando um ganho de desempenho.

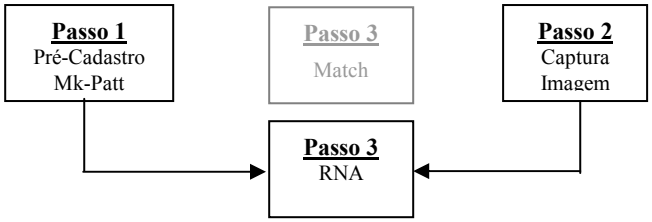


Figura 4. Substituição do Match (ArtoolKit) pela RNA para reconhecimento de marcadores válidos.

A RNA é implementada por uma fase de treinamento e outra de execução, portanto, é necessário diferenciar o que será feito na pré-execução e o que será feito na execução do ArtoolKit (Figura 5). A abordagem escolhida para a fase de treinamento foi a supervisionada, onde, é montando um conjunto de treinamento, formado por entradas e saídas desejadas, que é apresentado para a rede. Com isso a rede tem que ser capaz de ajustar os pesos (conexões), à medida que se apresenta o conjunto de treinamento, para que na fase de execução seja capaz de emitir saídas satisfatórias para entradas desconhecidas.

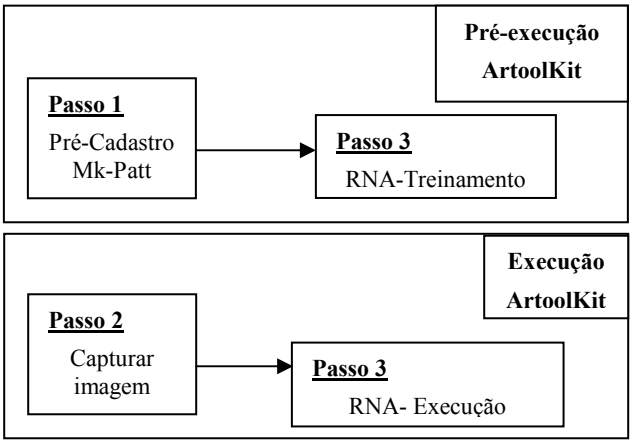


Figura 5. Fases de Pré-execução e de Execução do ArtoolKit.

4 Utilização da RNA no reconhecimento de marcadores válidos

Como exemplo, foi criado um cenário onde se utilizou três marcadores: Traço, Kaije e o Hiro. Desta forma, a rede tem 256 neurônios de entrada e três neurônios de saída. A técnica usada para treinamento foi a regra delta, o α utilizado é 0,0039 e a condição de parada é Erro Quadrático menor ou igual a 0,1. O α é a taxa de aprendizagem utilizada na rede, que na maioria das vezes é definida aleatoriamente. O α define o tamanho do passo para a aprendizagem da rede.

O Erro Quadrático é o somatório dos erros de cada neurônio de saída.

Na fase de pré-execução foram apresentados os marcadores da Figura 6 ao Mk-Patt que criou um arquivo, para cada marcador, com a matriz 16x16. Estes três arquivos formaram a base de treinamento que foi apresentado a RNA para que aprendesse. A RNA chegou ao Erro Quadrático de 0,09998 após 1745 ciclos. Uma amostra dos pesos encontrados está na figura 7. A equação a seguir mostra a fórmula do Erro Quadrático (Fausett, 1994).

$$E = \tfrac{1}{2} \sum (t^p - y^p)^2 \qquad (1)$$

Onde:
E = Erro quadrático
tp = Target de saída(saída esperada)
yp = Saída da rede

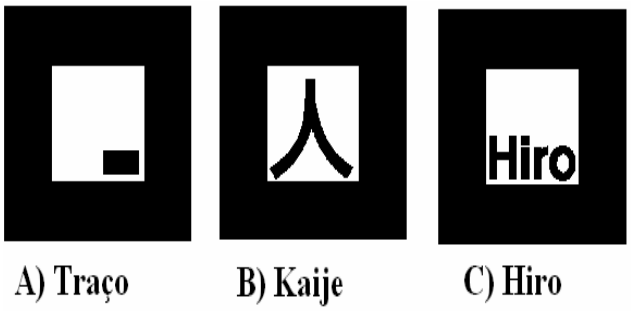


Figura 6. Marcadores utilizados no treinamento da RNA.

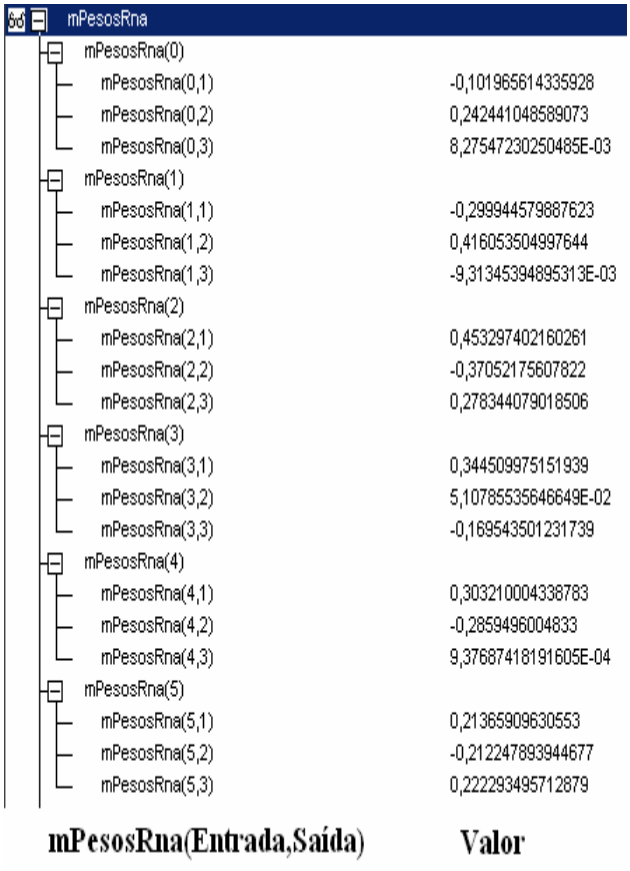


Figura 7. Alguns dos pesos ajustados durante o treinamento.

5 Resultados da análise de complexidade dos algoritmos

A análise de algoritmo é uma parte importante da teoria de complexidade de algoritmo, que provê estimativas teóricas sobre recursos necessários para qualquer problema computacional. Estas estimativas direcionam para uma melhor eficiência nos algoritmos (Cormen et. al.,2001; Knuth, 1997; Brassard e Bratley,1996).

Foram feitas as análises da complexidade dos algoritmos Match e RNA, e a análise de tempo de execução de cada linha. A figura 9 mostra o código do algoritmo Match do ARToolKit.

```
1:static    int    pattern_match(ARUint8
2:*data,   int  *code,  int  *dir,  double
3:*cf)
4: {
5:  int    input[768];
6:  int    i, j, k, l;
7:  int    ave, sum, res, res2;
8:  double datapow, sum2, max;
9:  sum = ave = 0;
10:    for(i = 0; i < 768 ;i++){
11:       ave += (255-data[i]);}
12:       ave /= 768;
13:    for(i = 0;i < 768; i++){
14:    input[i] = (255-data[i]) - ave;
15:    sum += input[i]*input[i];}
16:    datapow = sqrt((double)sum);
17:    if(datapow == 0.0){
18:        *code = 0;
19:        *dir  = 0;
20:        *cf   = -1.0;
21:        return -1;
22:     }
23:    res = res2 = -1;
24:    max = 0.0;
25:    k = -1;
26:for(l = 0; l < m; l++){
27:   k++;
28:        while(patf[k] == 0 ) k++;
29:    if( patf[k] == 2 ) continue;
30:   for( j = 0; j < 4; j++ ){
31:       sum = 0;
32:      for(i = 0; i < 768; i++)
33:       sum += input[i]*pat[k][j][i];
34:   sum2 = um/patpow[k][j]/datapow;
35:   if( sum2 > max )
36:   {max = sum2; res = j; res2 = k;}
37:    }
38: }
```

Figura 9. Algoritmo Match do ARToolKit (Kato e Billinghurst, 2000).

A tabela 1.0 mostra os cálculos de tempo de execução do algoritmo Match.

Tabela 1. Cálculo do tempo de execução do algoritmo Match. Sendo, o Custo da Linha o número de operações executadas na mesma e a Repetição a quantidade de vezes que a linha é executada.

Linha	Custo da Linha	Repetição	Tempo total da Linha (Custo Linha * Repetição)
10	2	769	1538
11	2	768	1536
12	1		1
13	2	769	1538
14	2	768	1536
15	2	768	1536
16	1		1
17	1		1
26	2	m + 1	2*m + 2
27	1	1*m	1*m
29	1	1*m	1*m
30	2	5*m	10*m
32	2	769*4*m	6152*m
33	2	768*4*m	6144*m
34	2	4*m	8*m
35	1	4*m	4*m
Tempo total de execução			**(12322*m) + 7689**

A figura 10 mostra o código do algoritmo RNA implementado para o reconhecimento de marcadores.

```
1:static void RNA(float Entradas[256],
2:float *&vSaida){
3:int  k,i;
4:double yin;
5:vSaida= new float[m];
6: for(k = 0; k < m; k++){
7: yin = 0;
8: for(i = 0;i <= 256;i++){
9:     yin=yin+(Entradas[i] * M[i][k]);
10:    }
11: if (yin >= 0.5) vSaida[k] = yin;
12: else vSaida[k] = -1;}
```

Figura 10. Algoritmo RNA.

A tabela 2.0 mostra os cálculos de tempo de execução do algoritmo RNA.

Tabela 2. Cálculo do tempo de execução do algoritmo Match. Sendo, o Custo da Linha o número de operações executadas na mesma e a Repetição a quantidade de vezes que a linha é executada.

Linha	Custo da Linha	Repetição	Tempo total da Linha (Custo Linha * Repetição)
6	2	m + 1	2*m + 2
8	2	257 * m	514 * m
9	2	256*m	512*m
11	1	m	1*m
Tempo total de execução			**(1029 * m) + 2**

As tabelas 1.0 e 2.0 são formadas por 4 colunas:

- Linha: que determina a linha que está sendo analisado do algoritmo;
- Custo da Linha: que especifica o custo baseado no número de operações executadas na linha. Sendo que, cada operação (adição, subtração, multiplicação, divisão, comparação) tem custo igual a um. Por exemplo, a linha 11 da figura 9, onde existe uma adição (ave +=) e uma subtração (255 – data[i]), tem 2 custos. Lembrando que, a atribuição não tem custo;
- Repetição: quantidade de vezes que a linha é executada;
- Tempo total da Linha: é o resultado da equação Custo da Linha * Repetição.

6 Conclusões

Baseado nos resultados obtidos, conclui-se que o algoritmo utilizando rede neural artificial reconhece os marcadores tal qual o módulo de reconhecimento usado pelo ARToolKit. Entretanto, apesar de ambos os algoritmos apresentarem uma complexidade de O(m), onde m representa número de marcadores, o algoritmo proposto tem tempo de execução bem menor que o algoritmo Match do ARToolKit. Assim, o algoritmo proposto provê um desempenho melhor no processo de reconhecimento dos marcadores, benefício que é de fundamental importância durante a execução de aplicações de RA em tempo real. Como trabalhos futuros serão feitos testes envolvendo um maior número de marcadores a fim de se medir o desempenho dos dois algoritmos. Além disso, serão feitos testes com marcadores em condições imperfeitas, para medir a qualidade do reconhecimento.

Referencias Bibliográficas

Azuma, R. (1993). Tracking Requirements for Augmented Reality, Communications of the ACM, 36(7):50-51, July.

Azuma, R. T. (1997). A Survey of Augmented Reality. Disponível em: http://www.cs.unc.edu/~azuma/ARpresence.pdf

Bajura, M. e Neumann, U. (1995). Dynamic Registration Correction in Video-Based Augmented Reality Systems, IEEE Computer Graphics & Applications, 15(5):52-60, Sept.

Boman, D.K. (1995). International Survey:Virtual Environment research, IEEE Computer, 28(6):57-65, June.

Brassard, G. e Bratley, P. (1996). Fundamentals of Algorithmics, Prentice Hall.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2001). Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill. ISBN 0-262-03293-7. Chapter 1: Foundations, pp.3–122.

Fausett, L.V. (1994). Fundamentals of Neural Networks: Architectures, Algorithms and Applications. Prentice-Hall, Englewood Cliffs, NJ.

Feiner, S. et al. (1993). Knowledge-Based Augmented Reality, Communications of the ACM, 36(7):52-62, July.

Kato, H. e Billinghurst, M. (2000). ARToolKit version 2.33: A software library for Augmented Reality Applications, 44p, Tutorial, November. Disponível em: <http://www.hitl.washington.edu/projects/ artoolkit/publications.htm>.

Kirner, C. (1999). Sistemas de Realidade Virtual. Disponível em: <http://www.dc.ufscar.br/~grv/tutrv/tutrv.htm>. Acesso em: 14 Jan. 2006.

Knuth, Donald (1997). The Art of Computer Programming, Volume 1: Fundamental Algorithms, Third Edition. Addison-Wesley. ISBN 0-201-89683-4. Section 1.2.11: Asymptotic Representations, pp.107–123.