An Adaptive Edge Detection Based Colorization Algorithm and Its Applications

Yi-Chin Huang Yi-Shin Tung Jun-Cheng Chen Sung-Wen Wang Ja-Ling Wu
Dept. of Computer Science and Information Engineering
Graduate Institute of Networking and Multimedia
National Taiwan University, Taipei, Taiwan
{ken, tung, pullpull, song, wjl}@cmlab.csie.ntu.edu.tw

ABSTRACT
Colorization is a computer-assisted process for adding colors to grayscale images or movies. It can be viewed as a process for assigning a three-dimensional color vector (YUV or RGB) to each pixel of a grayscale image. In previous works, with some color hints the resultant chrominance value varies linearly with that of the luminance. However, it is easy to find that existing methods may introduce obvious color bleeding, especially, around region boundaries. It then needs extra human-assistance to fix these artifacts, which limits its practicability. Facing such a challenging issue, we introduce a general and fast colorization methodology with the aid of an adaptive edge detection scheme. By extracting reliable edge information, the proposed approach may prevent the colorization process from bleeding over object boundaries. Next, integration of the proposed fast colorization scheme to a scribble-based colorization system, a modified color transferring system and a novel chrominance coding approach are investigated. In our experiments, each system exhibits obvious improvement as compared to those corresponding previous works.

Categories and Subject Descriptors
I.3.3 [**Picture/Image Generation**]: display algorithms.
I.4.2 [**Compression (Coding)**]: approximated methods.
I.4.6 [**Segmentation**]: edge and feature detection.

General Terms
Algorithms, Design, Experimentation.

Keywords
Colorization, color transfer, re-coloring, chrominance coding.

1. INTRODUCTION
The work of colorization needs two inputs: a grayscale image or video that needs to be colorized and the chrominance side-information. In our work, the chrominance side-information may be 1) some scribbled colors that user interactively paints [1], 2) another color image or video with similar color layouts or structures [2-3], and 3) color seeds that are extracted from the original color sources. The colorization procedure is the kernel of the whole system. In our work, the assisted edge information is generated first on the basis of grayscale contents, and different colorization methods are then applied according to the availability of different color information. In considering of the edge detection algorithm and the revised fast colorization method, our work behaves very different from previous works, and exhibits a superior performance. The proposed framework and the corresponding components are depicted in Figure 1.

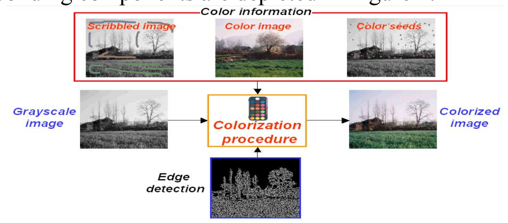

Figure 1. The proposed framework for colorization.

2. RELATED WORKS

2.1 Colorization Algorithms
Anat L. et al. [1] designed a user interactive interface for people to scribble some colors in an image, and then use an optimization process to colorize the remaining regions. They assumed that neighboring pixels with similar intensity (*Y*) would have similar chrominance values (*U and V*), and the chrominance values are then derived by minimizing the following equation.

$$J(U)=\sum_r\left(U(r)-\sum_{s\in N(r)} w_{rs}U(s)\right)^2 \tag{1}$$

where w_{rs} is a weighting function and the summation of weights applied for all pixels, *s*, near to *r* is set to one. The weight w_{rs} is larger when *Y(r)* is close to *Y(s)*. The following two weighting functions have been proposed and investigated:

$$w_{rs} \propto 1+\frac{1}{\sigma_r^2}(Y(r)-\mu_r)(Y(s)-\mu_r) \tag{2}$$

$$w_{rs} \propto e^{-(Y(r)-Y(s))^2/2\sigma_r^2} \tag{3}$$

2.2 Transferring Color to Grayscale Images
In [3], Tomihisa Welsh et al tried to transfer colors from a color image to a grayscale image (called source and target images, respectively). Assuming source and target images have similar color distribution and texture structure, chromaticity values are transferred by mapping each pixel from the grayscale image to the color image. The matching criterion is set on the basis of luminance value and the neighborhood statistics. To prevent incorrect color transfers, swatch windows are designed to mark similar regions between source and target images. The colors of corresponding swatches are transferred first, and are then used for colorizing the remaining pixels by applying an approach similar to the texture synthesis technique addressed in [4].

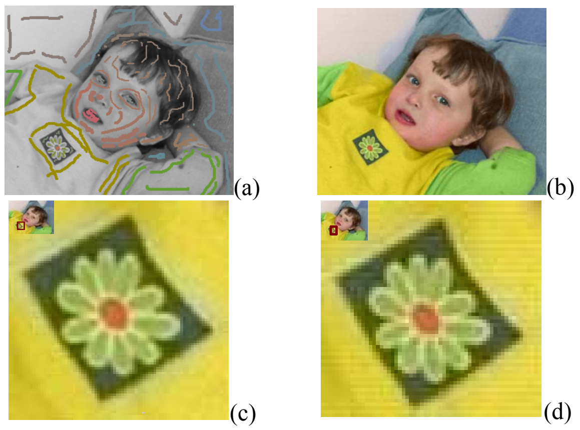

Figure 2. (a) Scribbled image, (b) source image, (c) the colorized image by using Eqn. (2), and (d) the colorized image by using Eqn. (4).

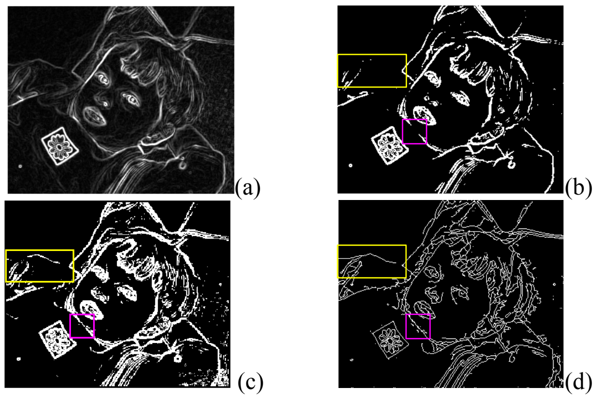

Figure 3. (a) Edge energy of the Sobel filter output, (b) Sobel edge detection with static threshold, (c) edge detection by the popular image processing software, Photoshop 7.0, and (d) the result of the proposed adaptive edge detection.

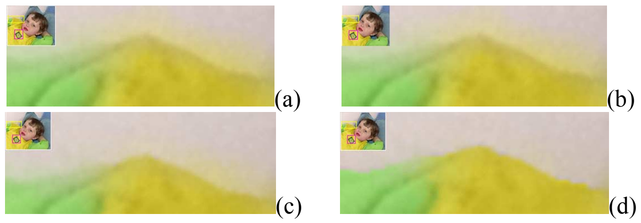

Figure 4. (a) Colorization output without edge detection, (b) with Sobel edge detection, (c) with adaptive edge detection, TH_{high}=90, TH_{low}=40, and (d) TH_{high}=80, TH_{low}=30.

2.3 Edge Detection

Edge detection is designate to identify the intensity discontinuity and mark out the object edges. It plays a very important role in our colorization system. Every grayscale image will be processed by the edge detection algorithm, and next by the colorization algorithm for each segmented region.

Edge detection is traditionally considered as either a convolution or correlation operation [5-7]. One can classify edge detection algorithms into two classes based on the support of the convolution kernel. When the kernel is with finite support, we call them mask-based edge detectors, such as the Sobel, Prewitt, Roberts and Kirsch detectors. The other group has the infinite support, such as two-dimensional Gaussian detectors. Marr-Hildreth and Canny detectors belong to this class. However, there are two drawbacks of these methods: 1) the edges extracted from the image are not very clear and slight intensity change may not be detected and 2) a static threshold must be decided. The choice of the threshold value is always made empirically.

3. THE ENHANCEMENT TECHNIQUES FOR COLORIZATION

3.1 Weighting Function

We retain to use the optimization algorithm as described in Section 2.1. However, we found that the two weighting functions (Eqns. (2) and (3)) do not always change the chrominance values proportional to the luminance similarities, so we propose a new weighting function in the following:

$$Wrs = 1/(1+|Y(r)-Y(s)|/(Var(r)+1)) \tag{4}$$

where r is a particular pixel, s is one of the neighboring pixels of r, and $Var(r)$ is the variance value of 3×3 windowed pixels centered on r. In our experiments, as shown in Figure 2, this weighting function outperforms those proposed in [1].

3.2 Adaptive Edge Detection Algorithm

From simulations, we found that some colorized regions are blurred because different object colors propagating and interlacing together, and as a result some pixels are painted with wrong colors. The wrong colorization usually occurs near the cross borders of two objects, and is visually obvious.

To fix this problem, an adaptive edge detection scheme is proposed. Firstly, we apply the Sobel filter with a high threshold, TH_{high}, to the input grayscale image, which generates our initial edge map. Four kinds of Sobel filters are adopted to detect horizontal, vertical, diagonal down-left, and opposite diagonal down-right edges, and the edge value, E_{Sum}, is derived as follows.

$$E_{Max} = Max(E_{Ver}, E_{Hor}, E_{Diag\_downleft}, E_{Diag\_downright}) \tag{5}$$

$$E_{sum} = \begin{cases} E_{Ver} + E_{Hor}, \text{if } E_{Max} == E_{Ver} \text{ or } E_{Max} == E_{Hor} \\ E_{Diag\_downleft} + E_{Diag\_downright}, \text{ otherwise} \end{cases} \tag{6}$$

Secondly, the pixels marked as edges will be extended with a lower threshold along the direction of the edge. The running threshold is decreased adaptively by a factor of 0.8 until a low bound, TH_{low}, is reached. After all edge pixels in the initial map have gone through this extension process, we get an extended edge map. However, the resultant edge data is still too rough to be used directly, because it may contain many noises and most edges are too thick. Thirdly, for each pixel in the extended edge map we find the local maximum among its 8 neighboring pixels by comparing their Sobel filtering outputs. Each pixel is either marked as local maximum or having its own maximum direction. Next, we start to link these local maximums for extracting edge skeletons. In this process, we sort all local maximums and try to link them to two of their neighboring maximums. In each trial, at most two liking edges are added. Finally, there are still some broken edges because of their Sobel filtering results are not large enough or lower than TH_{low}. We search in the extended edge map for a path to link these two extremities if they are not belonging to the same connected edge or of far distance along the edge. After performing the above steps, a clear edge map is exploited.

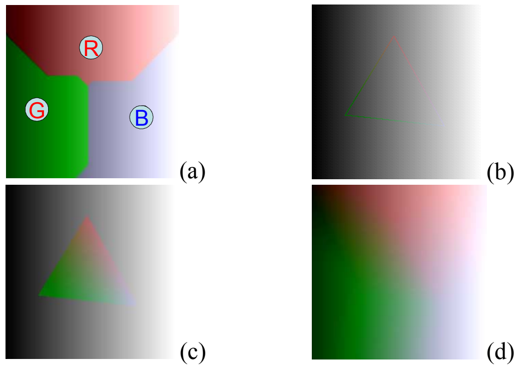

Figure 5. (a) Output after first iteration of the iterative algorithm, (b) output of edges colorized on the basis of our smooth equation, (c) output of the triangle colorized on the basis of the proposed approach and (d) output of colorization based on our non-iterative algorithm.

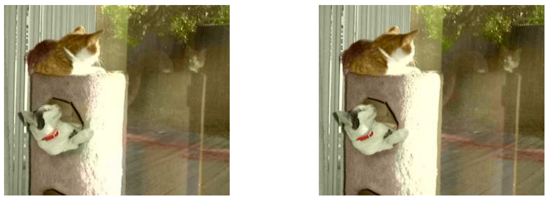

Figure 6. Output images of the iterative colorization algorithm (left: PSNR$_U$=36.3, PSNR$_V$=37.1) and the non-iterative colorization algorithm (PSNR$_U$=35.7, PSNR$_V$=38.4).

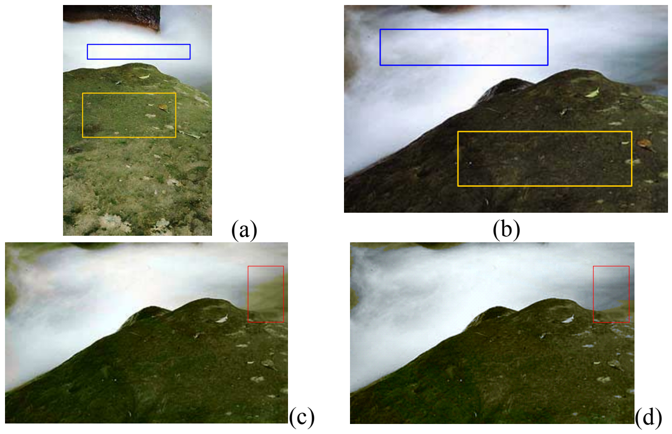

Figure 7. (a) Source image, (b) target image with its original colors, (c) the result of Tomihisa W.'s approach and (d) the result of our approach.

In our experiments, some unobvious edges can be successfully detected as shown in Figure 3. The improvement of colorization can be then obtained. An example is showed in Figure 4.

3.3 Iterative v.s. Non-Iterative

The most time-consuming part of colorization algorithm comes from hundreds of iterations for the optimization process, so we develop a non-iterative colorization algorithm to meet the requirements of some time-constrained applications. As shown in Figure 7(a), color discontinuity will occur if only one iteration is allowed. The artifact originates from the independent color flooding from two different colors. If we can connect those discontinuously colorized pixels first and assign suitable colors to pixels on the connecting path, the discontinuity effect will be minimized. Thus, we triangulate the pixels that have been assigned with some specific colors in the same object region by the Delauney triangulation algorithm. Then, we colorize the edge lines of the triangle by using the following equations.

$$U(\mathrm{Pr}) = {(a_1 \times U(P_1) + a_2 \times U(P_2))}\Big/{(a_1 + a_2)} \quad V(\mathrm{Pr}) = {(a_1 \times V(P_1) + a_2 \times V(P_2))}\Big/{(a_1 + a_2)} \tag{7}$$

$$D(P_1, P_2) = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \tag{8}$$

$$\begin{aligned} a_1 &= \{(Wr_{P1}/(Wr_{P1} + Wr_{P2}))*[1 + D(P_2,\mathrm{Pr})/D(P_1,P_2)]/2\} \\ &+(D(P_2,\mathrm{Pr})/D(P_1,P_2))*[1 - D(P_2,\mathrm{Pr})/D(P_1,P_2)]/2 \\ a_2 &= \{(Wr_{P2}/(Wr_{P1} + Wr_{P2}))*[1 + D(P_1,\mathrm{Pr})/D(P_1,P_2)]/2\} \\ &+[D(P_1,\mathrm{Pr})/D(P_1,P_2)]*[1 - D(P_1,\mathrm{Pr})/D(P_1,P_2)]/2 \end{aligned} \tag{9}$$

where P_1 and P_2 are the end points of an edge in a triangle, Pr is the pixel on the edge line between P_1 and P_2, $D(X, Y)$ is the distance function between the pixel pair X and Y, while W_{rp1} and W_{rp2} are the weighting functions derived from Eqn. (4).

Next, we initialize colors inside a triangle with its boundary colors. The pixel Pi at some horizontal line is colorized by interpolating its leftmost pixel P_1 and rightmost pixel P_2 on the triangle boundary, as formulated below.

$$\begin{aligned} U(P_i) &= U(P_1) \times D(P_2,P_i)/D(P_1,P_2) + U(P_2) \times D(P_1,P_i)/D(P_1,P_2) \\ V(P_i) &= V(P_1) \times D(P_2,P_i)/D(P_1,P_2) + V(P_2) \times D(P_1,P_i)/D(P_1,P_2) \end{aligned} \tag{10}$$

After all pixels within a triangle are initialized, we run the colorization along with the weighting function to include the effect of the luminance channel, as formulated in Eqns. (1) and (4). After this stage, the remaining pixels are colorized as usual in the next iteration. Finally, the remaining uncolored pixels are the edge pixels, which are given the color average of their neighbors.

We illustrate the results of our non-iterative algorithm step by step in Figure 5. Example results are shown in Figure 6. It is obvious from Figure 6 that both visual quality and PSNR are very close for the iterative and non-iterative algorithms.

4. COLOR TRANSFER

The most critical problem in color transfer applications is the existence of several regions with similar intensity distribution but having different colors. In order to solve this problem, edge-based segmentation is adopted to enhance the effective match. Firstly, the adaptive-threshold edge detection, as described in Section 3.2, is applied to the target image. Secondly, histogram remapping is processed for swatches between two images [3]. Thirdly, the edge map is involved in the color transfer of the remaining regions. Because the image has been segmented into several regions, we can successfully prevent incorrect matches from two different objects. The improvement is demonstrated in Figure 7.

5. COLORIZATION FOR VIDEO CODING

In this section, we present two chrominance coding schemes based on colorization, and realize them to extend the state-of-the-art video coding standard, H.264/MPEG-4 AVC.

5.1 Colorization with Color Seeds

Color seeds are a representative collection of colors coded with their locations and chrominance information. Seeds are treated as another input to our colorization algorithm. Similar to the non-iterative colorization for scribbled images, the algorithm comprises three preprocessing stages: edge detection for region segmentation, seed labeling for grouping seeds in the same region and color triangulation for a group of seeds. For those pixels on the triangle edge and inside triangles, Eqns. (7) and (10) are applied respectively, and the rest pixels outside triangles are colorized by Eqns. (1) and (4). In the encoding side, automatic seed generation algorithm, as shown in Figure 8, is devised. Seeds as well as the missing edge information are then differentially coded by using Exp-Golomb codes. In our experiments, the seed-based chrominance coding performs comparably to that of H.264/AVC. Figures 9 and 10 show some preliminary results. Although PSNR is lower, the visual quality is very similar and the bitrate is saved. Further, our algorithm eliminates color-bleeding artifacts and performs much better than H.264/AVC, especially for the cartoon sequences, as illustrated in Figure 10.

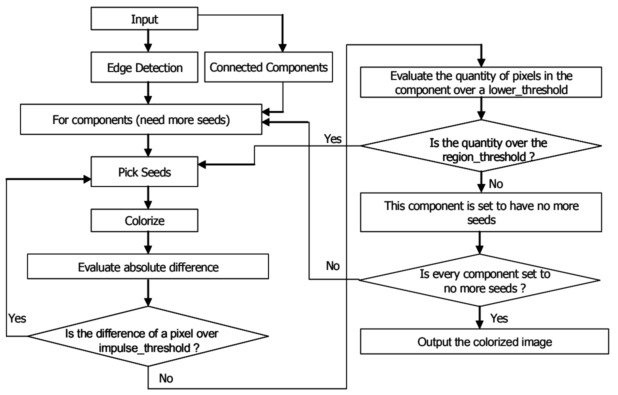
Input
Edge Detection
Connected Components
For components (need more seeds)
Pick Seeds
Colorize
Evaluate absolute difference
Is the difference of a pixel over impulse_threshold ?
Yes
No
Evaluate the quantity of pixels in the component over a lower_threshold
Is the quantity over the region_threshold ?
Yes
No
This component is set to have no more seeds
Is every component set to no more seeds ?
No
Yes
Output the colorized image

Figure 8. The flowchart of the color seed generation algorithm.

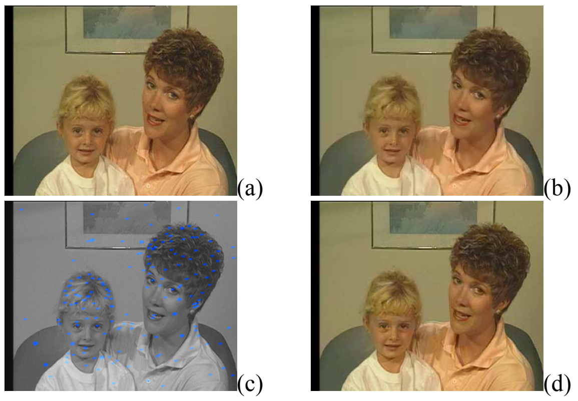

Figure 9. (a) Source frame, (b) H.264/AVC reconstruction result ($PSNR_U$=44.6, $PSNR_V$=43.6, bitrate=3764), (c) the selected color seeds (0.2% of total pixels) and (d) our colorization result ($PSNR_U$=40.8, $PSNR_V$=39.7, bitrate=2780).

5.2 Colorization Applied to Macroblock Level

The basic coding unit of H.264/AVC is a macroblock (MB), so we can apply the colorization process as a new chrominance prediction mode at the MB level. The color information is derived from the left- and the top-neighboring pixels in the previously coded MBs by referring edge information. The MB-level experiments are conducted and summarized in Table 1. Up to 0.5dB PSNR gain and 6.3% bitrate saving can be obtained.

6. CONCLUSION

We have presented a general framework for colorizing grayscale contents, which integrates the modified colorization method and the adaptive edge detection algorithm together. The proposal is then customized for three application instances. According to our experiments, in all cases the color bleeding artifacts are successfully eliminated. In the video coding applications, we also demonstrate a completely new usage of chrominance coding. Experimentally, it shows the best performance in the cartoon sequences. In summary, the proposed colorization system is so flexible that we can embed it into different applications easily.

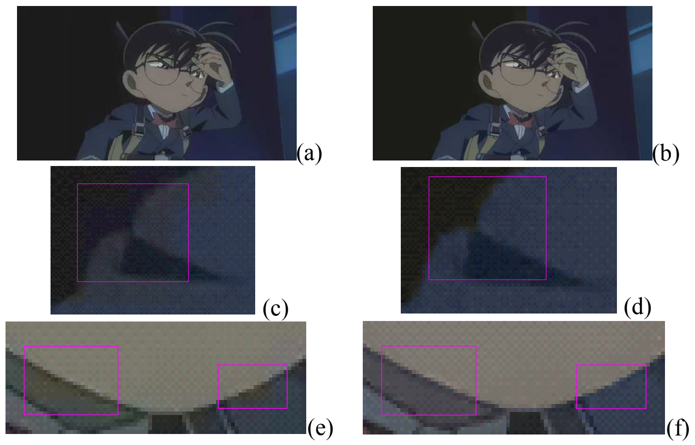

Figure 10. (a) H.264/AVC reconstruction ($PSNR_U$=43, $PSNR_V$=43, bitrate=3008), (b) result of colorization with 0.02% seeds ($PSNR_U$=43, $PSNR_V$=42, bitrate=2996), (c,e) enlarged parts of H.264/AVC reconstruction and (d,f) enlarged parts of colorization reconstructions.

Table 1. Average PSNRs and bitrates (I frame).

Sequence	Edge Threshold	H.264 Baseline (QP=36)			Proposed colorization-based prediction		
		$PSNR_U$	$PSNR_V$	Chroma Bitrate	$PSNR_U$	$PSNR_V$	Chroma Bitrate
Table tennis	160	39.99dB	38.78dB	3636bits	40.30dB	38.77dB	3442bits
Coastguard	160	43.04	44.17	1107	43.49	44.41	1190
Stefan	60	36.04	36.00	8796	36.07	36.01	8300
Football	120	37.28	38.57	4405	37.28	38.53	4126
Mother & daughter	80	41.47	42.33	1991	41.66	42.44	1968
Dancer	60	41.51	40.88	4025	42.01	40.96	3934
Salesman	60	38.96	40.05	3033	38.98	40.08	3014
Singer	60	35.43	36.16	11724	35.51	36.17	11156
Silent	120	37.63	38.95	4129	37.68	39.01	4018
Akiyo	60	39.91	40.73	4009	40.04	40.95	4001
Hall monitor	80	38.57	40.51	2644	38.71	40.54	2558
Foreman	80	39.50	41.55	2566	39.80	41.75	2562
Average		39.11	39.89	4338	39.30	39.97	4189
Improvement					0.18dB	0.07 dB	4.5%

7. REFERENCES

[1] Anat, L., Dani, L., and Yair, W. Colorization using optimization. *Proc. of SIGGRAPH* (Aug), 689-693.

[2] Erik, R., Michael, A., Bruce, G., and Peter, S. Color transfer between images. *IEEE Computer Graphics and Applications* (2001) 34-41.

[3] Welsh, T., Ashikhmin, M., and Mueller, K. Transferring color to greyscale images. *ACM Transactions on Graphics* 21, 3 (2002).

[4] Efros, A. A.and Freeman, W.T. Image Quilting for Texture Synthesis and Transfer. *Proc. of ACM SIGGRAPH 2001*, 341-346.

[5] Andreas, K., and Mongi, A. Detection and classification of edges in color images. *IEEE Signal Processing Magazine* (Jan 2005), 64-73.

[6] Dorin, C., and Peter, M. Robust analysis of feature spaces: color image segmentation. *Proc. of CVPR*, 1997.

[7] Gonzalez, R. C. and Wintz, P. *Digital Image Processing*. Addison-Wesley Publishing, Reading MA, 1987.