Super Precise Object Removal

Wentao Zhao
CS
518030910405
permanent@sjtu.edu.cn

Ren Zhou
CS
518030910408
zhouren@sjtu.edu.cn

Weixiong Lin
CS
518030910278
wx_lin@sjtu.edu.cn

***Abstract*—It is quite common for people's selfies to be ruined by unpleasant objects in background. Seeing that efficient object removal algorithms are in urgent need, we develop SPOR(Super Precise Object Removal). We propose an unprecedented algorithm to remove target object, using rough location of the target as input to generate high-resolution results. SPOR divides the task into finding target objects' contours, generating precise masks and removing target objects.**

The main idea is to calculate the energy map for pixels and interpret low energy pixels as unimportant ones. Prior energy of pixels is determined by image gradient. Additionally, we assign quite negative energy to pixels in mask region so that they will be preferentially removed. Then we construct a graph with each pixel as a node and each node pointing to several adjacent nodes in the following row. Then apply minimum cost maximum flow algorithm to find paths with lowest energy and remove pixels along.

To maintain as much information as possible, we generate the draft precise contour of target using GAN. But these contours may be unclosed, and it is closed ones that are required to generate masks, so we use heuristic search and contour overlaying to address this problem.

In comparison, We achieve better results(smoother contents and higher image quality) than some existing works like Telea and GANs which use traditional algorithms and deep learning method respectively. Extensive experiment results on object removal have demonstrated the effectiveness and universality of our method.

***Index Terms*—finding contours, object removal, high resolution, precise elimination**

I. Introduction

Women averagely spend 3 minutes to take a perfect selfie [1]. Unpleasant object in the background like passers-by is one reason for people to attempt again.

So the thirst of taking a perfect selfie has increased the importance of object removal techniques in recent years. Object removal algorithms aim to remove irrelevant background objects off pictures without side effects. Deep learning methods such as GAN usually cannot achieve precise and high-resolution removal results [2]. And traditional methods all require the precise mask of target object as input [3].

In this paper, we suggest a different pipeline called Super Precise Object Removal(SPOR) to perform high-resolution object removal. SPOR only takes a rough mask as input and returns high-resolution result. The ease of use makes SPOR practical to be applied on devices like smart phones. SPOR also achieves loosely coupling by dividing the task into 3 subtasks. Thus SPOR will evolve when any technique used in it make progress.

We show the successful application of the proposed method on difficult scenes. SPOR works perfectly independent of variant texture, brightness and colors. The universality makes SPOR highly competitive.

II. Related Works

We have summarized a number of methods for object removal and find that the most common way to remove unwanted objects is through image inpainting. That is to first crop the unwanted object from the image and then fill the cropped region by using other parts of the image. There are both traditional and modern ways of implementation and we show some examples below and state their weakness.

A. Modern Methods based on CNN

For instance, some modern methods train CNN to generate an arbitrary missing segment based on its surroundings in the image with an algorithm driven by context-based pixel prediction [5]. [6] adapts denoising auto-encoders to the task of blind image inpainting.

Drawbacks. The methods based on CNN is often data consuming, computationally intensive and also hard to train. They depend closely on the kind of images in the dataset, such as cats, dogs or human. That is to say, when we want to remove another type of object, the trained model may not satisfy our needs, which causes the problem of poor portability and inflexibility.

B. Some Other Traditional Methods

Paper [7] used an approach based on the "Navier-Stokes" equations for fluid dynamics, which used ideas from classical fluid dynamics to propagate isophote lines continuously from the exterior into the region to be inpainted. And paper [8] introduced an algorithm based on the fast marching method for level set applications, which was claimed to be simple to implement, fast, and able to produce similar results to more complex and slower known methods.

Drawbacks. The filled region produced by these methods is blurry and unnatural. And it crops larger region than necessary which impairs effect.

III. Method

Taking all the factors into consideration we have mentioned above, we propose our Object Removal methods(SPOR). We divide the problem into 3 subproblems: Contour Extraction,

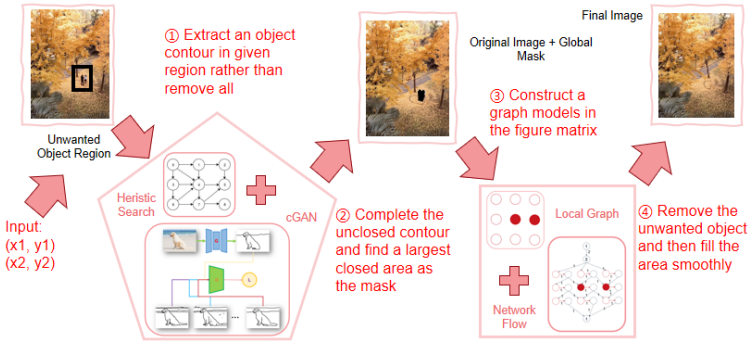


Fig. 1: SPOR

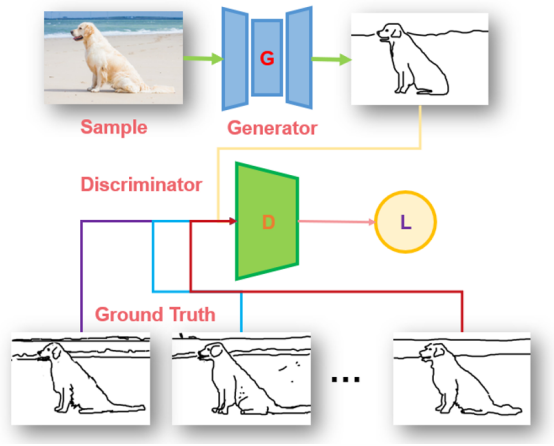


Fig. 2: The Training Process of cGAN

Mask Generation and Object Removal. And then we take appropriate approach step by step.

A. Contour Extraction

In many existing work, they give a rough region and remove all the pixels in it. But actually the unwanted object does not occupy so much space. Another way is to draw a mask to fit the position of the object image. Nevertheless, the contour is more likely to be irregular to sketch. We sincerely hope that our methods can be user-friendly and convenient to use so we just accept 2 coordinate points as the input of user, which are the top left and bottom right coordinates of a rough(can be very large) rectangular region. Then we automatically detect the main object and extract the contour.

Here we introduce conditional Generative Adversarial Network(cGAN) as one of our tools. This method was reproduced from the paper[5]. In this cGAN model, the generator network G takes input an color image x. At the same time, the discriminator network D is adversarially trained to distinguish the generator images and the ground truth target.For every image x_i, we have multiple target $y_i^{(1)}, y_i^{(2)}, \cdots, y_i^{(N)}$. The Min-Mean-loss function is defined as follow:

$$\mathcal{L}\left(x_i, y_i^{(1)}, \ldots, y_i^{(M_i)}\right) = \frac{\lambda}{M_i}\sum_{j=1}^{M_i}\mathcal{L}_{cGAN}\left(x_i, y_i^{(j)}\right) + \min_{j\in\{1,\ldots,M_i\}}\mathcal{L}_1\left(x_i, y_i^{(j)}\right)$$

Where M_i denotes the number of targets of x_i and $L_{c\_GAN}(\cdot)$ denotes the universal loss function of cGAN:

$$\mathcal{L}_{cGAN}(x, y, z) = \min_G \max_D \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]$$

Then we use our trained model to extract the contour of the images.

B. Mask Generation

We have got the rough contour of the unwanted object, but it is always a not closed contour. Since we want to remove the whole object precisely, what we need is a completed mask. So in this part we complete the contour and generate a mask for the unwanted object.

We propose two ways to get a closed mask: (I) Overlay the contour images which are produced by different training level model. (II) Using heuristic search to change inner pixels.

(I) The intrinsic motivation is overlaying different contour lines to make the contour be a closed region. We randomly choose cGAN models of different training iteration, then they will generate different contour lines. In some case, the lines generated from one model are exactly what is missing from another model. Then overlay them and we will complete this position's contour. The formulation are defined as follow:

$$C = 255 - Clip(\sum_{i=1}^{n}(255 - C_i), 0, 255)$$

Where C_i is the contour of one image, C is the final contour.

(II) Overlaying contours can not deal with all scenarios, so based on the final contour got by method(I), we using heuristic search to get the mask.

If a pixel is a part of the unwanted object, we regard it as an inner point, otherwise the outer point. So the intuition is that we extend 8 rays from one point to 8 directions, if the point is an inner point, it will have more cross lines with contour lines. And when user give two coordinate points to determine the region, our unwanted object is more likely to be in the center position of the region rather than corner. So based on these, we design our heursitic search algorithm.

For every pixel p in an image, we define l_{cnt} as the number of cross, lines, p_{cnt} as the number of cross pixels, p_{total} as the nubmer of total pixels in the contour. And (x_c, y_c) is the center of image, (x_p, y_p) is the pixel's position. Then we define the

heuristic function:

$$
\begin{aligned}
c_{weight} &= 0.5 - \frac{|x_p - x_c| + |y_p - y_c|}{H + W}, \\
p_{weight} &= p_{cnt}/p_{total}, \\
b_{weight} &= \alpha \cdot c_{weight} + (1 - \alpha) \cdot p_{weight}, \\
l_{weight} &= 5 + (l_{cnt} - 5) \times 0.8, \\
z &= b_{weight} + l_{weight} + \beta, \\
score &= tanh(z), \\
prob &= \left\{ \begin{array}{ll} 0 & , 0 < l_{cnt} \leq 4 \\ score & , 5 \leq l_{cnt} \leq 8. \end{array} \right\}
\end{aligned}
$$

We use l1-normal metric to determine the position weight and use tanh activation function to get the final probability. When we are search in the image, the white pixel will change to black with the probability we have calculated. By doing this, most of the inner pixel are changed to black to be a part of the contour, so we get a closed contour. Then we can using dfs search to get the maximum closed region. Changing all the pixels in this region to black and all other pixels in the image to white, we will finally get the black mask and it precisely cover the position of the unwanted region.

Algorithm 1 Heuristic Search

```
function GETSCORE(C)
    S: a (H, W) matrix
    for each pixel p(x, y) in C do
        lines, pixels = get_crosslines(C, p)
        if lines ≤ 4 then
            score = 0.0 S[x, y] = score
            continue
        end if
        pos_weight = 0.5 − (|x − c_x| + |y − c_y|)/(H + W)
        pix_weight = (pix_num)/(tot_pixel)
        bias_weight = α· pos_weight+(1 - α)· pix_weight
        line_weight = 5 + (lines − 5) × 0.8
        x = line_weight + bias_weight + β
        score = tanh(x)
        S[x, y] = score
    end for
    return S
end function

function COMPLETE(G,S)
    for each pixel(x, y) do
        if random_num ≤ S[x, y] then
            pixel(x, y) = black
        end if
    end for
    DFS(G) where pixel(x, y) is white and marks them
    DFS(G) where pixel(x, y) not be marked, record region
    Set the maximum region to be the black mask
    Set all other pixel to white return Mask
end function
```

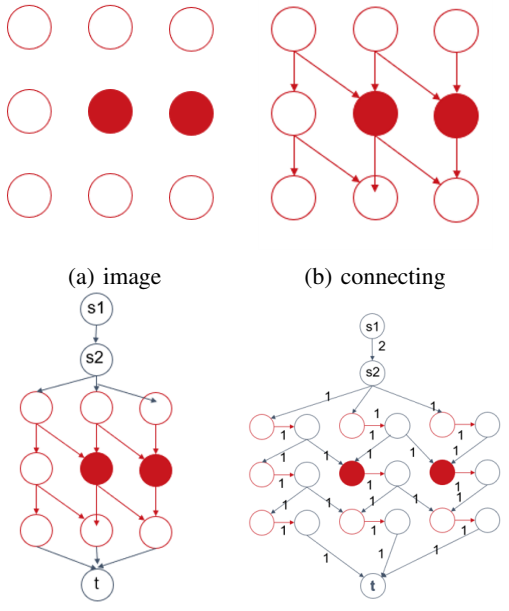
C. Object Removal

We realize that removing only pixels in mask region will result in different number of pixels in each row, making the photo ragged and unsmooth. So besides mask region, we have to remove some other pixels so that the number of pixels of each row is the same.

As to choosing what other pixels to remove, our basic idea is that pixels with lower energy are less important and removing them has smaller impact on holistic quality of the image. So we calculate the energy of each pixel and assign a quite negative value to pixels in mask region. We then remove pixels with lowest energy. To make sure that we remove the same number of pixels in each row, we find paths containing one pixel of each row and remove them.

In order to specify which path to remove, we assign costs to path, which is related to the sum of energy of nodes along that path, then find and remove paths with lowest energy. Minimum cost maximum flow algorithm has a great advantage in finding multiple paths with lowest costs, so we use it to find paths.

The following is the detail of turning picture and mask into a graph. We use an example to demonstrate this. Suppose there is a 3 by 3 image, and the red filled nodes are in mask region, as is shown in Figure 3a.



(a) image (b) connecting

(c) adding nodes (d) splitting

Fig. 3: Constructing graph

1) Set $k = \min(8,$width of image-1$)$, and connect each node to k nodes in their lower row. As is shown in figure 3b.
2) Add s1, s2 at the top and t at the bottom. Add edge (s1,s2). For all nodes v in the first row, add edge (s2,v). For all nodes v in the last row, add edge (v,t). As is shown in figure 3c.

3) To guarantee that different paths don't share the same node, split each node in original graph into two nodes and add an edge of unit capacity between them. Set the capacity of (s1,s2) to be min(width of mask,10)(the reason for this is explained later). Set the capacity of remaining edges to be 1. As is shown in figure 3d.

Next, use scharr operator(shown in figure 4) and convolution to calculate the energy of each pixel. We calculate the energy of red, green and blue channels separately and add them together.

$$E_{X_{red}} = \|S_x * X_{red}\| + \|S_y * X_{red}\|$$
$$E_{X_{green}} = \|S_x * X_{green}\| + \|S_y * X_{green}\|$$
$$E_{X_{blue}} = \|S_x * X_{blue}\| + \|S_y * X_{blue}\|$$
$$E_X = E_{X_{red}} + E_{X_{green}} + E_{X_{blue}}$$

-3	0	3
-10	0	10
-3	0	3

(a) S_x

-3	-10	-3
0	0	0
3	10	3

(b) S_y

Fig. 4: Scharr Operator

To remove nodes in mask region, we multiply their energy by -1000. Ergo, when we find paths with lowest energy later, they will be incorporated into certain paths.

Next, we define the cost of edges. For horizontal edges, their existence is just to guarantee that different paths don't share nodes, so we can assign 0 to their costs. For other edges, their cost is equal to half of the sum of the energy of their ends. That, is:

$$\text{cost}((u,v)) = \begin{cases} 0 & \text{if (u,v) is a horizontal edge} \\ \frac{E(u)+E(v)}{2} & \text{otherwise} \end{cases}$$

Now that we have complete constructing graph, we can use minimum cost maximum flow algorithm to find multiple paths with lowest costs. To preserve image quality, we cannot find and remove too many paths at once. Instead, we have to find some, remove, calculate the energy of nodes again, then repeat this process until all the nodes in mask region are removed. So we set the number of paths to be found in each running of our algorithm to be min(width of mask, 10). Instead of using vanilla BFS search to find augmenting paths, we use a quicker algorithm called shortest path faster algorithm(SPFA) to find paths. The whole idea is shown in Algorithm 2

After removing all the pixels in mask region, we need to add the same number of pixels to each row to preserve the shape of the picture. According to our energy interpretation of pixels, low energy means unimportant and has small impact on holistic image quality. Therefore, it is natural to come up with the idea of finding paths with lowest energy in the remaining picture and copy and insert them into it. And it turns out that this method can add pixels to picture without distorting the image or lowering the quality in most scenarios.

Algorithm 2 Finding paths

```
function SPFA(G,s)
    for each vertex v≠s in V(G) do
        d(v)=∞
    end for
    d(s)=0
    put s into queue Q
    while Q is not emtpy do
        pop the top element u from Q
        for each edge (u,v) in E(G) do
            if d(u)+w(u,v)¡d(v) then
                d(v)=d(u)+w(u,v)
                if v is not in Q then
                    add v into Q
                end if
            end if
        end for
    end while
end function

function MAX FLOW(G)
    Initially f(e) = 0 for all e in G, Sϕ, f0
    while there is an s1 − t path in residual graph G_f do
        Use SPFA to find path P
        Add P to S
        Agment along P
        Update f and G_f
    end while
    return S, f
end function
```

As for time complexity, denote the number of nodes and edges as n and m. Denote the width of mask as w. Constructing graph is O(mn). Calculating energy map is also O(mn). Calculatin the width of mask is O(mn). We construct graph, calculate energy and width of mask for about $\frac{w}{10}$ times, so this part is O(wmn). The average time complexity of Each run of SPFA is O(m), and we run it around w times, so this part is O(wm).

To conclude, our total time complexity is O(wmn).

IV. Experiment

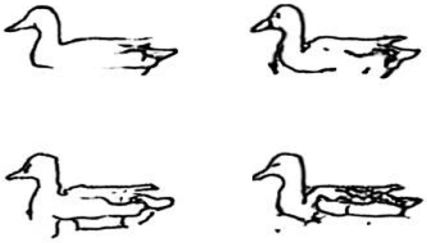
A. Contour Extraction

We train the cGAN model for 1000 iterations and save 9 models of different training levels. We give a color image, they will generate contour of different styles.

iterations	200	400	600	800	1000
G	54.43	38.08	27.22	20.59	15.24
D	2.09	1.07	0.65	0.18	0.16

TABLE I: G loss and D loss of Training Iteration

(a) The Rough Region of Color Image



(b) Contours Generated by Different Models

B. Mask Generation

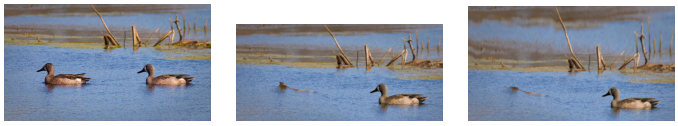
We use contour overlaying technique and heuristic search, we can convert some contours of one image to a full mask step by step.

C. Object Removal

The constructed graph computed the minimum cost maximum flow paths and delete them in the pictures. The target object is removed, but its size shrinks 8b. So we use the same technique to calculate minimum cost maximum flow paths again. And insert pixels into images with linear interpolation. The images swells up to the initial size in this process.

Fig. 6: Overlay Different Contours

Fig. 7: Heuristic Search and Generate Mask



(a) Raw image (b) Object deleted (c) Resized back

Fig. 8: Heuristic Search and Generate Mask

(a) Raw scene (b) GAN's output (c) SPOR's output

Fig. 9: Contrastive Outputs

D. Results

To examine the advantages of SPOR, we show contrastive outputs of SPOR and GANs. The figure 9 suggests that SPOR offer better performance than GANs' fine-tuning model from ImageNet.

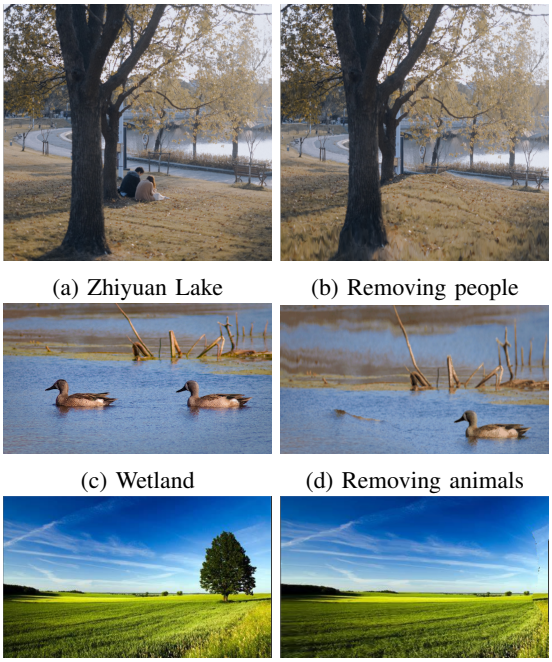
Due to the square shape of provided raw mask, GAN generates an image with object removed but mosaic produced. The blurred square in 9b is marked in red. Such low resolution output is not qualified for practical use. Let alone various objects that aren't included in GAN's training, which could be worse. And SPOR achives nature result in the comparison, removing object and maintaining high-resolution. The reason not to test SPOR on different metrics is that, they are not suitable for object removal tasks. Because metrics like MSE and PSNR all focus on the similarity between raw pictures and removal ones. But object removal is meant to replace all pixels under masks, heading just the opposite way. But SPOR's advantages are too obvious that you can tell by observation and intuition.

In regrad of universality, the author of 'Removal of Background People' [2] only provides limited demos, which are all human removal. But SPOR can transfer to different scenes with no obstacles. As shown in figure 10, SPOR holds stable performance for a variaty of backgrounds and targets.

V. Contribution

In SPOR, our major contributions are:

- We jettison the end-to-end system structure and decompose the problem into three subproblems: contour extraction, mask generation and object removal. The greatest advantage of this is low coupling, so that if there is a new efficient way to solve one of our subproblems, we



(a) Zhiyuan Lake (b) Removing people

(c) Wetland (d) Removing animals

(e) Grassland (f) Removing plants

Fig. 10: SPOR in various scenes

can easily introduce it to one part of our system without modifying other sections.

- Instead of removing pixels in region boxed by users directly, we attempt to find a more accurate contour of object, resulting in removing less pixels and higher image quality.
- As for finding contours, GAN cannot give closed contours. We creatively design a heuristic search algorithm to obtain a closed contour. Combined with contour overlaying, we can detect objects of various categories. So our method is portable and versatile, not restricted to certain types of objects.
- We adopt the minimum cost maximum flow algorithm which takes both image semantics and performance into consideration. Due to this algorithm, we can find and remove multiple paths at one time, which helps us boost our system's speed and performance.

VI. Conclusion

In our Super Precise Object Removal project, we unprecedentedly use minimum cost maximum flow algorithm to remove unwanted objects. Our basic idea is to calculate the energy of each pixels and regard pixels with low energy as unimportant. We assign quite negative energy to pixels we want to remove. Then use minimum cost maximum flow algorithm to find vertical paths on which pixels have low energy and remove them. After removing the object, we find paths with lowest energy in the remaining picture and copy and insert them into the image to preserve shape. Using this traditional algorithm frees us from tedious training and is quite computationally efficient. For most pictures, we can thoroughly remove unwanted objects without making the pictures blurry.

To further improve the image quality, we did more work than simply removing object circled by users. We try to obtain more accurate contour of object so as to decrease the number of pixels to be removed. We trained a GAN neural network to produce draft contours of object, but this may not be a closed contour so it cannot be directly filled and used. To address this problem, we use contour overlaying combined with an innovative heuristic search algorithm to turn the unclosed contour into a closed one.

In comparison, We achieve better results(smoother contents and higher image quality) than some existing works like Telea, which uses other traditional algorithms.

References

[1] https://www.stylist.co.uk/life/how-many-attempts-to-take-a-selfie-research-survey-social-media/183237

[2] Y. Li and X Li, "Removal of Background People Using Object Detection and Inpainting", 2015.

[3] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," ACM Trans. Graphics, vol. 26, no. 3, pp. 10.1-10.9, 2007.

[4] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[5] D.Pathak, P.Krhenbhl, J.Donahue, T.Darrell, and A.A.Efros, "Context encoders: Feature learning by inpainting," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[6] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," Advances in neural information processing systems, 2012.

[7] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001.

[8] A. Telea, "An image inpainting technique based on the fast marching method," J. Graphics, GPU, Game Tools, 2004.

[9] K. Elissa, "Title of paper if known," unpublished.

[10] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[11] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[12] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.