# REPORT DOCUMENTATION PAGE



| 1. AGENCY USE ONLY (Leave blank) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | | FINAL 30 Sep 90 - 29 Sep 92 |

**4. TITLE AND SUBTITLE**
"INTERACTIVE GRID GENERATION ON SMALL COMPUTERS" (U)

**5. FUNDING NUMBERS**
65502F
3005/A1

**6. AUTHOR(S)**
Drs Ning Lu, Kooroosh Nikfetrat & Peter Eiseman

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Program Development Corporation of Scarsdale, Inc.
300 Hamilton Ave, Suite 409
White Plains, NY 10601

**8. PERFORMING ORGANIZATION REPORT NUMBER**
AFOSR-TR- 93 0022

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
AFOSR/NM
110 Duncan Ave Suite B115
Bolling AFB DC 20332-0001

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
F49620-90-C-0080



**11. SUPPLEMENTARY NOTES**

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**
Approved for public release;
Distribution unlimited

**12b. DISTRIBUTION CODE**
UL

**13. ABSTRACT** (Maximum 200 words)

Progress has been made in both algebraic and elliptic grid generation. (1) The control point form (CPF) of algebraic grid generation has been improved in two aspects. First, new blending functions are developed which allow a computer software user to specify the locations of control surfaces arbitrarily. Second, a new implementation scheme is used in the CPF to recapture the clustering feature of given grids. (2) An effective technique of curvature control in elliptic grid generation has been developed and implemented to control the grid point distribution along curved boundaries.



**14. SUBJECT TERMS**

**15. NUMBER OF PAGES**
51

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | SAR |

# Interactive Grid Generation on Small Computers

Ning Lu
Koorosh Nikfetrat
Peter Eiseman

Program Development Corporation of Scarsdale, Inc.
300 Hamilton Avenue, Suite 409
White Plains, New York 10601

November 1992

Final Report for SBIR Phase II Contract No. F49620-90-C-0080 in Period September 1990 - September 1992

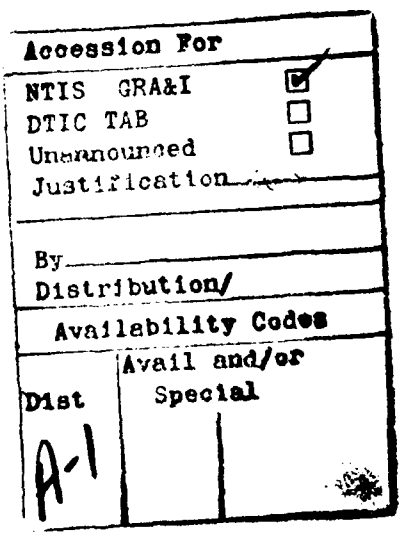
## ABSTRACT

Progress has been made in both algebraic and elliptic grid generation. (i) The control point form (CPF) of algebraic grid generation has been improved in two aspects. First, new blending functions are developed which allow a computer software user to specify the locations of control surfaces arbitrarily. Second, a new implementation scheme is used in the CPF to recapture the clustering feature of given grids. (ii) An effective technique of curvature control in elliptic grid generation has been developed and implemented to control the grid point distribution along curved boundaries.

## TABLE OF CONTENTS







## CHAPTER I. INTRODUCTION

The control point form (CPF) of algebraic grid generation for a 2D (or 3D) grid results from a combination of the multisurface transformation [1,2] and the transfinite interpolation [3]. The multisurface transformation can be thought of as a new kind of curve generation technique, like many other kinds of curve generation methods. From the point of view of a computer software user, the user would like to specify the locations of control points at his or her choice. This flexibility for a user requires that a set of blending functions can be constructed for any given set of control points. Previously, the blending functions used in the multisurface transformations have been developed [1,2]. However, the locations of control points in the physical space are determined *after* the blending functions are constructed. As a result, the locations of the control points in the physical space cannot be chosen arbitrarily. Put the problem into the context of grid generation within a 2D physical region, certainly we can provide a set of control points for a user. From the point of view of a computer software user, he or she may still want to insert or remove one or more control curves. In order to meet the user's needs, we should develop a new set of blending functions which would allow a user to choose the locations of control curves arbitrarily. Once we have such a set of blending functions, we can easily handle the problem of allowing a user to add and delete one or more control curves. In Chapter II of this report, we present our results on new blending functions–both $C^1$- and $C^2$-continuity–which allow arbitrary specification of the locations of the control points.

The blending functions we mentioned in the paragraph above are all local interpolation functions. This locality gives the CPF the capability of changing or modifying a given grid locally. The general procedure goes like this: A user has a grid which has been generated in one of other methods. Then, the user wants to improve the quality of his grid locally using the CPF method. Usually, this requires first regenerating the user's grid using the CPF method. After regeneration, the user can move one or more control points to change the grid locally. In previous application of the CPF to regenerating a given grid, it was found that, for a given grid with clustering along one or more boundary curves, the grid regenerated using the CPF method often fails to recapture the general clustering feature of the given grid near concave regions (cf. Figs. 3.1–3.3). To make the application of the CPF method successful, we need first to find out the reason of the failure and then to correct it. It turns out that the reason of the failure was in the usual implementation of connecting the continuous world of mathematics (analytic formula) and the discrete world of numerical computations (usually encountered in numerical grid generation). For this problem, we do not need any new blending functions. For any set of blending functions, whether old or new, we can use them to recapture the general clustering pattern of any given grid along one or more boundary curves, since they all satisfy the uniformity condition [1]. In Chapter III we report our progress made in this area and our new implementation of the control point form of algebraic grid generation. Also in Chapter III, the grids generated using both old and new implementation methods of the CPF are presented. From those illustrations of 2D grids, one can easily see the dramatic improvements in the concave regions for a grid with clustering.

Chapter IV of this report deals with improving the quality of grids generated elliptically. It describes an effective technique for controlling the tendency of elliptic systems to

pull grid points away from concave regions of the grid. This tendency is responsible for the poor grid resolution near concave regions, exhibited by elliptic grids. The discussion in Chapter IV takes a close and detailed look at the discrete form of the governing equations and isolates the terms which pull the points in various directions. In other words, Chapter IV describes a technique for improving grid quality by controlling the effects of curvature of the boundaries. The term used to designate this technique in this report is "curvature control." Results from a program which implements the above technique are presented and show improvements in grid quality over grids generated without the help of curvature control.

## CHAPTER II. MULTISURFACE TRANSFORMATION

In this chapter, we report our new blending functions developed in the Phase II work. This chapter is organized in the following way. In Section 2.1, we give the general properties of the multisurface transformation. In Section 2.2 we review previous results on $C^1$–continuity blending function. In Section 2.3 we construct new the $C^1$–continuity blending functions. In Section 2.4 we present old results on $C^2$–continuity blending functions. In Section 2.5 we develop new $C^2$–continuity blending functions.

### 2.1. General Formalism

To state the results mathematically some notation is needed. For this purpose, let $\mathbf{P}_1$, $\mathbf{P}_2$, ..., $\mathbf{P}_N$ be the given sequence of points in a 2D space; let $r$ be the curve parametrization; let $\mathbf{P}(r)$ be the position at $r$ along the desired curve; let $r_1$, $r_2$, ... $r_{N-1}$ be the successive parametric locations to interpolate the directions of $(\mathbf{P}_2 - \mathbf{P}_1)$, $(\mathbf{P}_3 - \mathbf{P}_2)$, ..., $(\mathbf{P}_N - \mathbf{P}_{N-1})$; and let $\psi_1(r)$, $\psi_2(r)$, ..., $\psi_{N-1}(r)$ be the corresponding interpolation functions which successively separate each direction by assuming a non-zero value at the associated location while vanishing at the remaining locations for interpolation. In two dimensions $\mathbf{P}_k = (x_k, y_k)$ and $\mathbf{P}(r) = (x(r), y(r))$. With this notation the curve is given by

$$\mathbf{P}(r) = \mathbf{P}_1 + \sum_{k=1}^{N-1} \frac{G_k(r)}{G_k(r_{N-1})}(\mathbf{P}_{k+1} - \mathbf{P}_k), \tag{2.1.1}$$

where

$$G_k(r) = \int_{r_1}^{r} \psi_k(x)dx, \quad k = 1, 2, ..., N-1. \tag{2.1.2}$$

To witness the basic specifications mentioned above, it is easy to check the end conditions $\mathbf{P}(r_1) = \mathbf{P}_1$ and $\mathbf{P}(r_{N-1}) = \mathbf{P}_N$ and the interpolatory condition that $d\mathbf{P}(r_k)/dr$ is in the direction of $(\mathbf{P}_{k+1} - \mathbf{P}_k)$ for each $k$ from 1 to $N-1$. In the context of coordinate generation for two- or three-dimensional regions, the endpoints become boundary surfaces and the interior points becomes control surfaces. For this reason the transformation generated by curves of the above form has been called a *multisurface transformation.*

Corresponding to the $N$ points $\mathbf{P}_1$, $\mathbf{P}_2$, ..., $\mathbf{P}_N$, we label their parametric values as $b_1$, $b_2$, ..., $b_N$, which are in an increasing sequence

$$b_1 < b_2 < ... < b_N. \tag{2.1.3}$$

The values of $b_1$, $b_2$, ..., $b_N$ can be specified or determined by, say, $b_k = \mathbf{P}_k \cdot \tau$ (we will discuss how to choose these $b$'s values in Chapter III). Because of relation (2.1.3), the $N-1$ parametric values for $N-1$ derivative directions are also in an increasing sequence,

$$r_1 < r_2 < ... < r_{N-1}. \tag{2.1.4}$$

The two sets of parametric values are coincide at two ends of the curve,

$$b_1 = r_1, \qquad b_N = r_{N-1}, \tag{2.1.5}$$

The interpolation function $\psi_k(r)$ in Eq. (2.1.2) satisfies a cardinality condition,

$$\psi_k(r_j) = \delta_{jk}, \qquad j,k = 1,2,...,N-1. \tag{2.1.6}$$

An important condition for the curve (2.1.1) is a uniformity condition, which states that, when projected on to a vector $\tau$, the curve becomes a simple linear curve in the parametric space, $\mathbf{P}(r) \cdot \tau = r$. Let

$$C_k = b_{k+1} - b_k, \qquad k = 1,2,...,N-1, \tag{2.1.7}$$

the uniformity condition is expressed mathematically as

$$r = r_1 + \sum_{k=1}^{N-1} \frac{G_k(r)}{G_k(r_{N-1})} C_k. \tag{2.1.8}$$

Taking derivative with respect to $r$ on both sides of Eq. (2.1.8), we obtain

$$1 = \sum_{k=1}^{N-1} \frac{\psi_k(r)}{G_k(r_{N-1})} C_k. \tag{2.1.9}$$

The cardinality condition (2.1.6) makes the summation collapse to only one term, yielding

$$C_k = G_k(r_{N-1}), \qquad k = 1,2,...,N-1. \tag{2.1.10}$$

Substituting Eq. (2.1.10) back into Eqs. (2.1.8) and (2.1.9), we simplify the uniformity condition to

$$r = r_1 + \sum_{k=1}^{N-1} G_k(r), \tag{2.1.11}$$

$$\sum_{k=1}^{N-1} \psi_k(r) = 1. \tag{2.1.12}$$

The actual values of the parameters $r_k$ and $b_k$ are related in some way. If one first chooses $r_k$'s, then $b_k$'s are determined by Eqs. (2.1.7) and (2.1.10). On the other hand, if one first is given $b_k$'s, then one has to decide $r_k$'s by using Eq. (2.1.10) and other choices. We will discuss these "other choices" in Sections 2.3 and 2.5. In any case, since the two sets of parametric locations are coincide in two ends [see Eq. (2.1.5)], the sum of all intervals in terms of one set of parametric locations must be equal to that in terms of the other set of parametric locations. In other words, if we define

$$h_k = r_{k+1} - r_k, \qquad k = 1,2,...,N-2, \tag{2.1.13}$$

then we have a restriction for the two sets of intervals,

THIS
PAGE
IS
MISSING
IN
ORIGINAL
DOCUMENT

In the rest of this chapter, we report $C^0$ and $C^1$ interpolants $\psi_k$, i.e., $C^1$ and $C^2$ blending functions $G_k$. We first consider $C^1$ continuity for blending functions $G_k(r)$'s in Sections 2.2 and 2.3: The interpolation functions $\psi_k(r)$'s are piecewise continuous themselves, whereas the blending functions $G_k(r)$'s are continuous up through first derivatives. Then, we consider $C^2$ blending functions in Sections 2.4 and 2.5. The interpolation functions $\psi_k(r)$'s are continuous up through first derivatives, whereas the blending functions $G_k(r)$'s are continuous up through second derivatives.

## 2.2. Previous Results on $C^1$-Continuity Blending Functions

The simplest local interpolants belong to the class $C^0$ of continuous functions. The interpolation functions $\psi_k$ consists of two pieces for $k = 2, 3, ..., N-2$. Each of the interpolants $\psi_1$ and $\psi_{N-1}$ for the two end points is made up of only one piece. An illustration of continuous piecewise *linear* interpolation functions is plotted in Fig. 2.1.

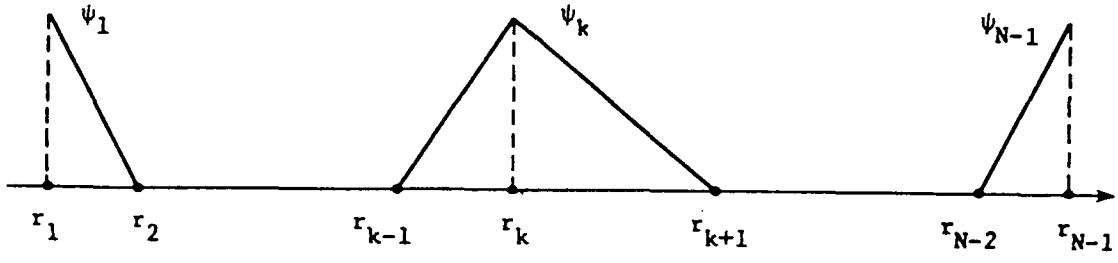



**Figure 2.1.** Continuous piecewise linear interpolation functions.

In the region $r_k < r < r_{k+1}$, only two blending functions are non-zero, and the uniformity condition (2.1.12) reduces to

$$\psi_k(r) + \psi_{k+1}(r) = 1, \qquad r_k < r < r_{k+1}, \qquad k = 1, 2, ..., N-2. \tag{2.2.1}$$

For arbitrary positions of $r_k$, the interpolation functions must be of the following form,

$$\psi_1(r) = \begin{cases} 1 - f_1(x_1), & \text{for } r_1 \le r \le r_2, \\ 0, & \text{for } r_2 \le r \le r_{N-1}, \end{cases} \tag{2.2.2a}$$

$$\psi_k(r) = \begin{cases} 0, & \text{for } r_1 \le r \le r_{k-1}, \\ f_{k-1}(x_{k-1}), & \text{for } r_{k-1} \le r \le r_k, \\ 1 - f_k(x_k), & \text{for } r_k \le r \le r_{k+1}, \\ 0, & \text{for } r_{k+1} \le r \le r_{N-1}, \end{cases} \tag{2.2.2b}$$

for $k = 2, 3, ..., N-2$,

$$\psi_{N-1}(r) = \begin{cases} 0, & \text{for } r_1 \le r \le r_{N-2}, \\ f_{N-2}(x_{N-2}), & \text{for } r_{N-2} \le r \le r_{N-1}, \end{cases} \tag{2.2.2c}$$

where

$$x_k = \frac{r - r_k}{r_{k+1} - r_k} = \frac{r - r_k}{h_k}, \qquad k = 1, 2, ..., N-2. \tag{2.2.3}$$

In Ref. [1], simple piecewise linear interpolation functions were used (see Fig. 2.1),

$$f_k(x) = x. \tag{2.2.4}$$

Substituting Eqs. (2.2.2) and (2.2.4) into Eq. (2.1.2) and completing the integrals, we obtain the blending functions

$$G_1(r) = \begin{cases} h_1(x_1 - \frac{1}{2}x_1^2), & \text{for } r_1 \le r \le r_2, \\ \frac{1}{2}h_1, & \text{for } r_2 \le r \le r_{N-1}, \end{cases} \tag{2.2.5a}$$

$$G_k(r) = \begin{cases} 0, & \text{for } r_1 \le r \le r_{k-1}, \\ \frac{1}{2}h_{k-1}x_{k-1}^2, & \text{for } r_{k-1} \le r \le r_k, \\ \frac{1}{2}h_{k-1} + h_k(x_k - \frac{1}{2}x_k^2), & \text{for } r_k \le r \le r_{k+1}, \\ \frac{1}{2}(h_{k-1} + h_k), & \text{for } r_{k+1} \le r \le r_{N-1}, \end{cases} \tag{2.2.5b}$$

for $k = 2, 3, ..., N-2$, and

$$G_{N-1}(r) = \begin{cases} 0, & \text{for } r_1 \le r \le r_{N-2}, \\ \frac{1}{2}h_{N-2}x_{N-2}^2, & \text{for } r_{N-2} \le r \le r_{N-1}. \end{cases} \tag{2.2.5c}$$

A graphical illustration of the blending functions $G_k(r)$ is drawn in Fig. 2.2. Each function increases monotonically from 0 to its maximum value $G_k(r_{N-1})$. According to Eq. (2.1.10), these maximum values give the intervals of the control points $\mathbf{P}_k$ in the parametric space,

$$C_1 = \frac{1}{2}h_1, \tag{2.2.6a}$$

$$C_k = \frac{1}{2}(h_{k-1} + h_k), \qquad k = 2, 3, ..., N-2, \tag{2.2.6b}$$

$$C_{N-1} = \frac{1}{2}h_{N-2}. \tag{2.2.6c}$$

Since the coefficients represented by Eqs. (2.2.5) are known functions, the curve of equation (2.1.1) depends only upon the sequence of points $\mathbf{P}_1$, $\mathbf{P}_2$, ..., $\mathbf{P}_N$.

The values of $r_k$'s can be chosen arbitrarily. The simplest choice is to choose a unit spacing in the interpolation points, $r_k = k$. This causes both the interpolation functions and their integrals $G_k$ to be translations of a function about the origin. The overall consequence is a simply stated curve definition. In an analytical form the origin-centered interpolation function is given by

$$\psi(x) = \begin{cases} 1 - |x|, & \text{for } -1 \le x \le 1, \\ 0, & \text{otherwise}, \end{cases} \tag{2.2.7}$$

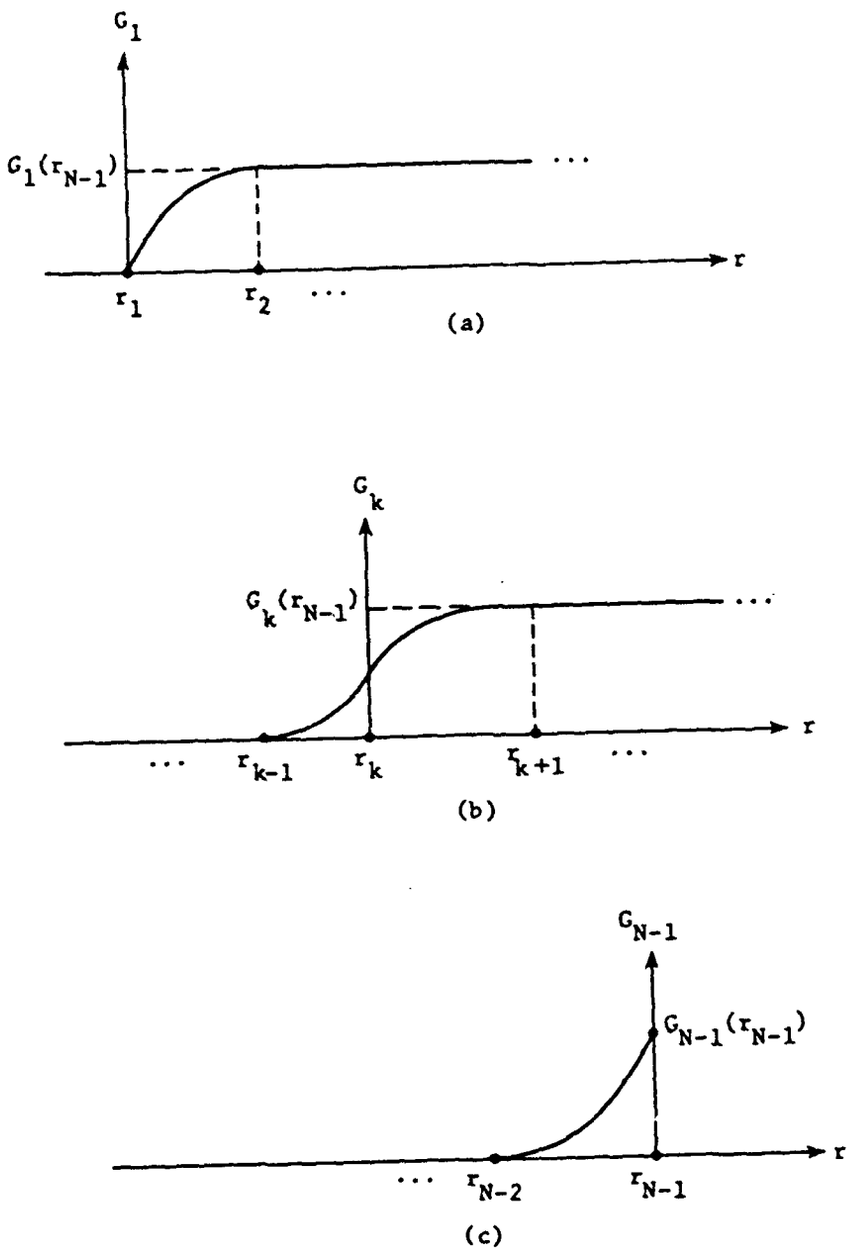



**Figure 2.2.** Integrals of the $C^0$ interpolation functions from Fig. 2.1.

and its integral is given by

$$\Omega(x) = \begin{cases} 0, & \text{for } x \le -1, \\ \frac{1}{2}(x+1)^2, & \text{for } -1 \le x \le 0, \\ 1 - \frac{1}{2}(x-1)^2, & \text{for } 0 \le x \le 1, \\ 1, & \text{for } x \ge 1. \end{cases} \tag{2.2.8}$$

In this simple case, the coefficients in Eq. (2.1.1) are given by

$$\frac{G_1(r)}{G_1(r_{N-1})} = 2\Omega(r-1) - 1, \tag{2.2.9a}$$

$$\frac{G_k(r)}{G_k(r_{N-1})} = \Omega(r-k), \quad k = 2, 3, ..., N-2, \tag{2.2.9b}$$

$$\frac{G_{N-1}(r)}{G_{N-1}(r_{N-1})} = 2\,\Omega(r-N+1). \tag{2.2.9c}$$

Also, the intervals $C_k$ are of the form of "half one one ... one half,"

$$C_1 = C_{N-2} = \frac{1}{2}, \tag{2.2.10a}$$

$$C_k = 1, \qquad k = 2, 3, ..., N-2. \tag{2.2.10b}$$

Besides piecewise linear interpolation function (2.2.4) or (2.2.7), it is certainly true that one can construct various other $C^0$ continuity interpolation functions. For example, one can have the following piecewise trigonometric interpolation function,

$$\psi(x) = \begin{cases} \cos^2(\frac{1}{2}\pi x), & \text{for } -1 \le x \le 1, \\ 0, & \text{otherwise}, \end{cases} \tag{2.2.11}$$

which is continuous up through first derivatives. Thus, the corresponding blending functions $G_k(r)$ are continuous up through second derivatives.

The scheme discussed in this section determines the two sets of parametric locations $(r_k, h_k)$ and $(b_k, C_k)$ in the following order;

$$(r_k, h_k) \rightarrow (\psi_k(r), G_k(r)) \rightarrow (b_k, C_k). \tag{2.2.12}$$

From the viewpoint of application, this kind of scheme has a disadvantage: If a computer software user wants to specify the parametric locations $b_k$ at his or her choice, then it does not tell us how to determine $r_k$ and $C_k$. Besides, it is not possible to find a corresponding set of $r_k$'s in certain situations. For example, since $h_k > 0$ for all $k$, we see from Eqs. (2.2.6) that $C_1 < C_2$ always. Thus, this scheme is not capable of handling the case in which $C_1 \ge C_2$. More flexible scheme is needed. We will present a new scheme in the next section.

## 2.3. New Results on $C^1$-Continuity Blending Functions

For any given set of $b_k$'s, the question is how to determine $r_k$'s. Our way of choosing $r_k$'s is

$$h_1 = C_1 + \frac{1}{2}C_2, \tag{2.3.1a}$$

$$h_k = \frac{1}{2}(C_k + C_{k+1}), \qquad k = 2, 3, ..., N-3, \tag{2.3.1b}$$

$$h_{N-2} = \frac{1}{2}C_{N-2} + C_{N-1}. \tag{2.3.1c}$$

To consolidate Eqs. (2.3.1) into a single equation, we introduce

$$\tilde{C}_k = \frac{1}{2}C_k, \qquad k = 2, 3, ..., N-2, \tag{2.3.2a}$$

$$\tilde{C}_k = C_k, \qquad k = 1, N-1, \tag{2.3.2b}$$

and thus have a simple form for Eqs. (2.3.1),

$$h_k = \tilde{C}_k + \tilde{C}_{k+1}, \qquad k = 1, 2, ..., N-2. \tag{2.3.3}$$

The piecewise linear function described in Section 2.2 does not work here, since $G_k(r_{N-1}) = \frac{1}{2}(h_{k-1} + h_k)$ in Eq. (2.2.3b) does not satisfy Eq. (2.1.10) in general. More general and flexible interpolation functions are needed. Starting from the first interval $r_1 < r < r_2$ for the interpolation functions, according to Eqs. (2.1.2) and (2.1.10), we have to require that the area under the interpolation function $\psi_1(r)$ is

$$\int_{r_1}^{r_2} \psi_1(r)dr = G_1(r_2) = G_1(r_{N-1}) = C_1 = \tilde{C}_1. \tag{2.3.4}$$

Because of Eqs. (2.3.4) and (2.3.1a), this makes the area under the interpolation function $\psi_2(r)$ within the first interval is

$$G_2(r_2) = \int_{r_1}^{r_2} \psi_2(r)dr = \int_{r_1}^{r_2} [1 - \psi_1(r)]dr = h_1 - C_1 = \frac{1}{2}C_2 = \tilde{C}_2. \tag{2.3.5}$$

Then, we examine the second interval $r_2 < r < r_3$. In order to satisfy Eq. (2.1.10), we have to require that the area under the interpolation function $\psi_2(r)$ within the second interval is

$$\int_{r_2}^{r_3} \psi_2(r)dr = G_2(r_3) - G_2(r_2) = G_2(r_{N-1}) - \frac{1}{2}C_2 = \frac{1}{2}C_2 = \tilde{C}_2. \tag{2.3.6}$$

Proceeding in this way, we find that, for the $k$th ($k = 1, 2, ..., N-2$) interval, the requirements are

$$G_k(r_{k+1}) - G_k(r_k) = \int_{r_k}^{r_{k+1}} \psi_k(r)dr = \tilde{C}_k, \tag{2.3.7a}$$

$$G_{k+1}(r_{k+1}) = \int_{r_k}^{r_{k+1}} \psi_{k+1}(r)dr = \tilde{C}_{k+1}. \tag{2.3.7b}$$

Equation (2.3.7b) can be rewritten as

$$G_k(r_k) = \tilde{C}_k, \qquad k = 2, 3, ..., N-1. \tag{2.3.8}$$

However, $G_1(r_1) = 0$ is an exception.

We have mentioned above that the piecewise linear interpolation function will not satisfy Eqs. (2.3.7) in general. In order to satisfy Eqs. (2.3.7), one more parameter (freedom) should be allowed in the form of the interpolation function. One choice is to use a quadratic interpolation function,

$$f_k(x) = x + A_k x(x-1). \tag{2.3.9}$$

Equation (2.3.9) has been set in a convenient form which satisfies the two requirements $\psi_k(r_k) = 1 - f_k(0) = 1$ and $\psi_k(r_{k+1}) = 1 - f_k(1) = 0$. Substituting Eqs. (2.2.2) and (2.3.9) into Eqs. (2.3.7), it is straight forward to find that

$$A_k = 3(\tilde{C}_k - \tilde{C}_{k+1})/h_k, \qquad k = 1, 2, ..., N-2. \tag{2.3.10}$$

The sign of $A_k$ depends on whether $C_k > C_{k+1}$ or $C_k < C_{k+1}$. When $C_k = C_{k+1}$, the coefficient $A_k$ vanishes, $A_k = 0$. The quadratic interpolation function $f_k(x)$ is not positive definite within the range $0 < x < 1$. For example, when $A_k > 1$, i.e., when $C_k > 2C_{k+1}$, the function $f_k = -(A_k - 1)^2/4A_k$ is negative at $x = (A_k - 1)/2A_k < \frac{1}{2}$. To ensure the interpolation functions to be positive definite, we can choose another form

$$f_k(x) = x^{m_k}, \tag{2.3.11}$$

which has the same end values $f_k(0) = 0$ and $f_k(1) = 1$ as the $f_k(x)$ given in Eq. (2.3.9). Substituting Eqs. (2.2.2) and (2.3.11) into Eqs. (2.3.7), we obtain

$$m_k = \frac{\tilde{C}_k}{\tilde{C}_{k+1}}, \qquad k = 1, 2, ..., N-2. \tag{2.3.12}$$

Integrating the interpolation functions $\psi_k(r)$, we obtain the blending functions

$$G_1(r) = \begin{cases} h_1 x_1 - H_1(x_1), & \text{for } r_1 \le r \le r_2, \\ \tilde{C}_1, & \text{for } r_2 \le r \le r_{N-1}, \end{cases} \tag{2.3.13a}$$

$$G_k(r) = \begin{cases} 0, & \text{for } r_1 \le r \le r_{k-1}, \\ H_{k-1}(x_{k-1}), & \text{for } r_{k-1} \le r \le r_k, \\ \tilde{C}_k + h_k x_k - H_k(x_k), & \text{for } r_k \le r \le r_{k+1}, \\ 2\tilde{C}_k, & \text{for } r_{k+1} \le r \le r_{N-1}, \end{cases} \tag{2.3.13b}$$

$$G_{N-1}(r) = \begin{cases} 0, & \text{for } r_1 \le r \le r_{N-2}, \\ H_{N-2}(x_{N-2}), & \text{for } r_{N-2} \le r \le r_{N-1}, \end{cases} \tag{2.3.13c}$$

where

$$H_k(x) = h_k \int_0^x f_k(x)dx. \tag{2.3.14}$$

For the quadratic interpolation function (2.3.9), we get

$$H_k(x) = (2.2\tilde{C}_{k+1} - \tilde{C}_k)x^2 + (\tilde{C}_k - \tilde{C}_{k+1})x^3, \qquad k = 1, 2, ..., N-2; \tag{2.3.15}$$

whereas for the positive-definite interpolant (2.3.11), we have

$$H_k(x) = \tilde{C}_{k+1} x^{h_k/\tilde{C}_{k+1}}, \qquad k = 1, 2, N-2. \tag{2.3.16}$$

With "half one one ... one half" intervals for $C_k$'s,

$$2C_1 = C_2 = C_3 = ... = C_{N-2} = 2C_{N-1}, \tag{2.3.17}$$

we find that, all $\tilde{C}_k$'s and $h_k$'s are equal

$$\tilde{C}_k = \frac{1}{2}h_k = \frac{1}{2}C_2, \qquad k = 1, 2, ..., N-1. \tag{2.3.18}$$

In this special case, for the quadratic interpolant (2.3.9), we have $A_k = 1$, whereas for the positive definite interpolant (2.3.11), we get $m_k = 1$. Thus, for both of them, we obtain

$$H_k(x) = \tilde{C}_{k+1}x^2 = \frac{1}{2}h_k x^2, \tag{2.3.19}$$

and recover the interpolation and blending functions reviewed in Section 2.2. In this special case, Eqs. (2.2.6) and (2.3.1) give the same relation between $C_k$'s and $h_k$'s. In general, Eqs. (2.3.1) are different from Eqs. (2.2.6). In Fig. 2.3, we present an example of relation between $h_k$'s and $C_k$'s using Eqs. (2.2.6) and (2.3.1).

To summary the result of this section, we stress that the scheme used here proceeds in the following order:

$$(C_k, b_k) \rightarrow (r_k, h_k) \rightarrow (\psi_k(r), G_k(r)). \tag{2.3.20}$$

## 2.4. Previous Results on $C^2$-Continuity Blending Function

For blending functions $G_k(r)$ with $C^2$ continuity, the corresponding interpolation functions $\psi_k(r)$ are of $C^1$ continuity. It has been shown in Ref. [2] that, in order to admit the possibility of uniformity and to avoid any unspecified flat spots, a local interpolation function which is not close to either boundary point must be non-zero over a minimum of 4 intervals. In other words, the interpolation function $\psi_k(r)$ is non-zero in the region of $r_{k-2} < r < r_{k+2}$ $(k = 3, 4, ..., N-3)$. In Fig. 2.4 the general form of the interpolation functions $\psi_k$ are displayed. As a result, in the region $r_k < r < r_{k+1}$, only (up to) four blending functions are non-zero, and the uniformity condition (2.1.12) reduces to

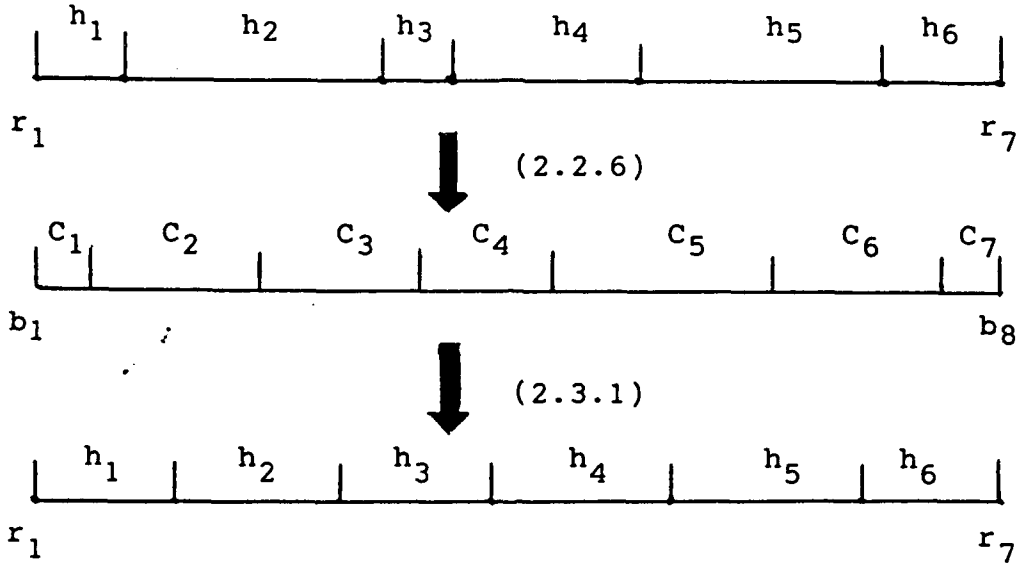



**Figure 2.3.** Relations between $h_k$'s and $C_k$'s for a case of $N = 8$.

$$\psi_{k-1}(r) + \psi_k(r) + \psi_{k+1}(r) + \psi_{k+2}(r) = 1, \quad r_k < r < r_{k+1}, \quad k = 2, 3, ..., N-3, \tag{2.4.1a}$$

$$\psi_1(r) + \psi_2(r) + \psi_3(r) = 1, \qquad r_1 < r < r_2, \tag{2.4.1b}$$

$$\psi_{N-3}(r) + \psi_{N-2}(r) + \psi_{N-1}(r) = 1, \qquad r_{N-2} < r < r_{N-1}. \tag{2.4.1c}$$

It is easier to construct the interpolation functions starting from their first-order derivative, since piecewise continuous functions can be used. As mentioned in Section 2.1, the interpolation functions must satisfy Eq. (2.1.12). Consequently, the first-order derivative of all interpolation functions adds up to zero exactly in the whole region,

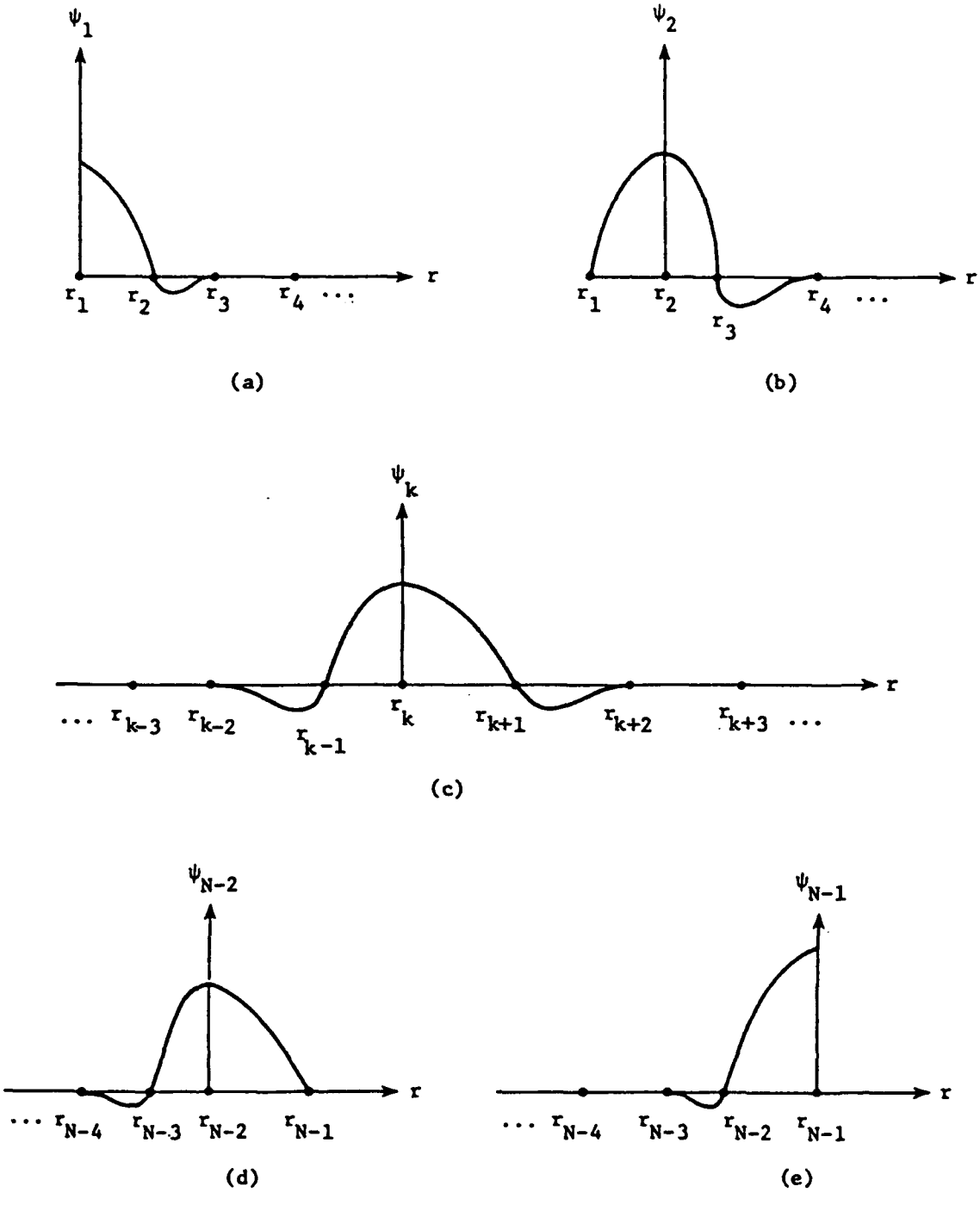



**Figure 2.4.** Schematic form of the simplest $C^1$ interpolation functions for the partition $r_1 < r_2 < \ldots < r_{N-1}$.

$$\sum_{k=1}^{N-1} \psi_k'(r) = 0. \tag{2.4.2}$$

In Ref. [2] $\psi_k'(r_k) = 0$ for $k = 2, 3, ..., N-2$ have been used. Upon the evaluation at the partition points, the uniformity condition was found to become

$$\psi_1'(r_1) + \psi_2'(r_1) = 0, \tag{2.4.3a}$$

$$\psi_{N-2}'(r_{N-1}) + \psi_{N-1}'(r_{N-1}) = 0, \tag{2.4.3b}$$

for end points and

$$\psi_{k-1}'(r_k) + \psi_{k+1}'(r_k) = 0, \tag{2.4.3c}$$

for $k = 2, 3, ..., N-3$.

In Ref. [2], the same scheme as described in Section 2.1 was used, i.e., the scheme (2.2.12) was used to first construct the local interpolation and blending functions and then find the parametric intervals $C_k$ and locations $b_k$. Specializing to the simple case of equal intervals $r_k = k$ (i.e., $h_k = 1$) and $\alpha_k = b_k = \frac{1}{2}$, $c_k = d_k = \frac{7}{4}$, the blending functions in Ref. [2] should reduce to

$$\frac{G_k(r)}{G_k(r_{N-1})} = \begin{cases} \Omega(r-k)/24, & \text{for } k = 2, 3, ..., N-2, \\ \Omega(r-k)/25, & \text{for } k = N-2, \\ [\Omega(r-k)+1]/25, & \text{for } k = 2, \\ [\Omega(r-k)+\Omega(r-N)]/11, & \text{for } k = N-1, \\ [\Omega(r-k)+\Omega(r)-37]/11, & \text{for } k = 1, \end{cases} \tag{2.4.4}$$

where

$$\Omega(x) = \begin{cases} 0, & \text{for } x \le -2, \\ -2(x+2)^3, & \text{for } -2 \le x \le -\frac{3}{2}, \\ 6(x+1)^3 + 6(x+1)^2 - 1, & \text{for } -\frac{3}{2} \le x \le -1, \\ 10(x+1)^3 + 6(x+1)^2 - 1, & \text{for } -1 \le x \le -\frac{1}{2}, \\ -14x^3 + 24x + 12, & \text{for } -\frac{1}{2} \le x \le \frac{1}{2}, \\ 10(x-1)^3 - 6(x-1)^2 + 25, & \text{for } \frac{1}{2} \le x \le 1, \\ 6(x-1)^3 - 6(x-1)^2 + 25, & \text{for } 1 \le x \le \frac{3}{2}, \\ -2(x-2)^3 + 24, & \text{for } \frac{3}{2} \le x \le 2, \\ 24, & \text{for } x \ge 2. \end{cases} \tag{2.4.5}$$

These blending functions are illustrated in Fig. 2.5. In order to satisfy Eq. (2.1.14), it is straightforward to check that the intervals must be

$$C_3 = C_4 = ... = C_{N-3} = 1, \tag{2.4.6a}$$

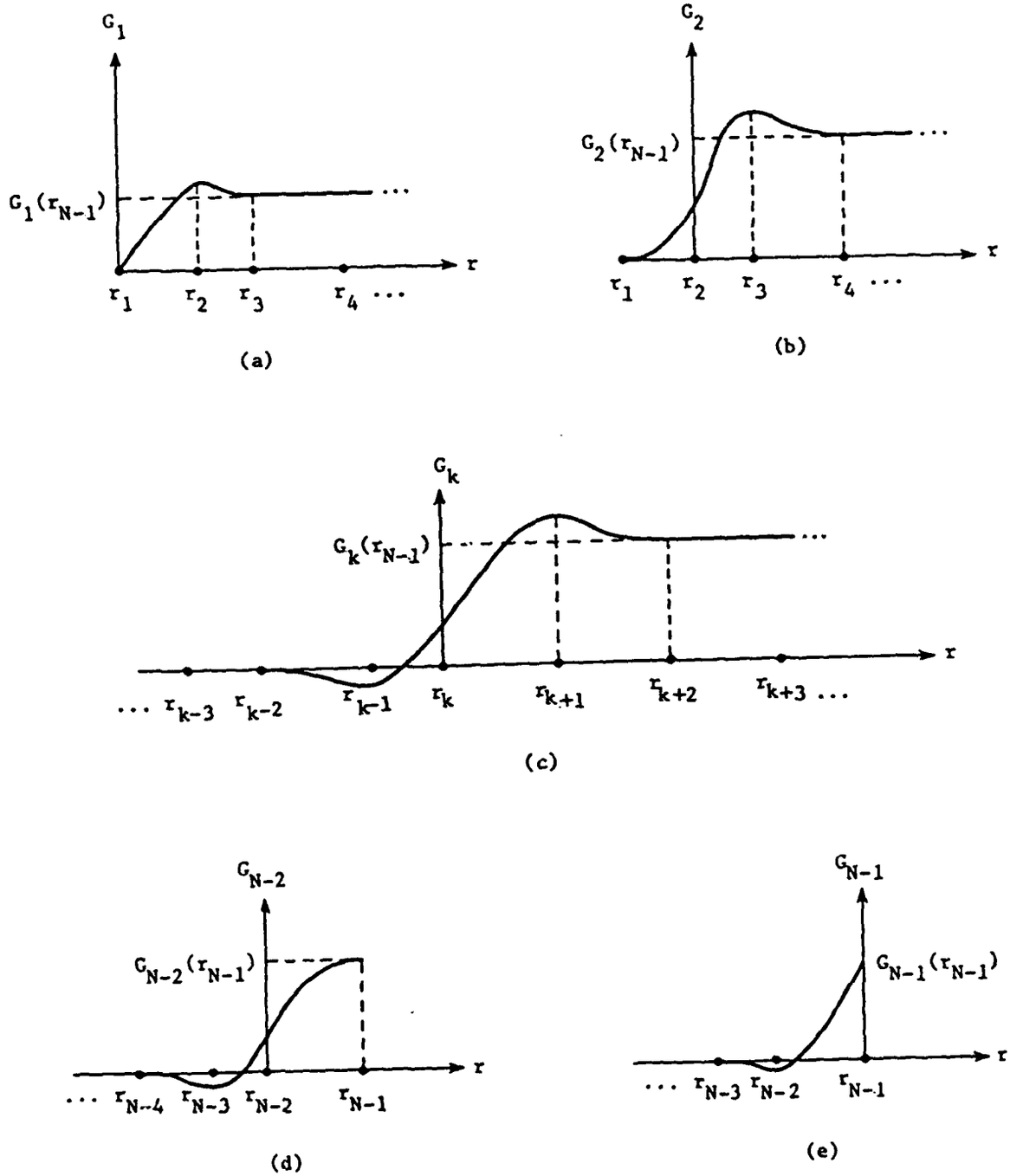



**Figure 2.5. Integrals of the $C^1$ interpolation functions.**

$$C_1 = C_{N-1} = \frac{11}{24}, \quad C_2 = C_{N-2} = \frac{25}{24}, \tag{2.4.6b}$$

which are not exactly of the "half one one ... one half" spacing given in Section 2.2 for $C^1$ blending function.

## 2.5. New Results on $C^2$-Continuity Blending Functions

Similar to the situation in Section 2.3, questions here are: For an arbitrarily given set of parametric locations $b_k$, how to determine (or choose) the values of $h_k$, and how to construct the interpolation functions. It is obvious that there are many ways to accomplish these things. But we want to them having certain properties. As in Section 2.3, we want to have $G_k(r_k) = \frac{1}{2}C_k$ for blending functions not close to end points. We also want to the ratios of 4 areas under the interpolation function $\psi_k(r)$ to be independent of the index $k$ and to be $\eta : 1 : 1 : \eta$. As will be evident in the following [cf. Eq. (2.5.16)], the value of $\eta$ must be within $-1 < \eta < 0$. [For blending functions (2.4.5), $\eta = -1/13$.] Based on these considerations, for a given set of $C_k$, we first introduce

$$\tilde{C}_k = \frac{C_k}{2(1+\eta)}, \qquad k = 3, 4, ..., N-3, \tag{2.5.1a}$$

$$\tilde{C}_k = \frac{C_k}{2+\eta}, \qquad k = 2, N-2, \tag{2.5.1b}$$

$$\tilde{C}_k = \frac{C_k}{1+2\eta}, \qquad k = 1, N-1, \tag{2.5.1c}$$

$$\tilde{C}_0 = \tilde{C}_1, \qquad \tilde{C}_N = \tilde{C}_{N-1}. \tag{2.5.1d}$$

Then, we choose

$$h_k = \tilde{C}_k + \tilde{C}_{k+1} + \eta(\tilde{C}_{k-1} + \tilde{C}_{k+2}), \qquad k = 1, 2, ..., N-2. \tag{2.5.2}$$

The choice of $\eta$ should lead to all $h_k$ positive, which requires that, for all $k$, $|\eta| < (\tilde{C}_k + \tilde{C}_{k+1})/(\tilde{C}_{k-1} + \tilde{C}_{k+2})$, i.e.,

$$|\eta| < \text{Min}\{(\tilde{C}_k + \tilde{C}_{k+1})/(\tilde{C}_{k-1} + \tilde{C}_{k+2})\}. \tag{2.5.3}$$

In order to write down the area requirements in terms of a unified form, we let

$$\psi_k(r) = \tilde{\psi}_k(r), \qquad k = 2, 3, ..., N-2 \tag{2.5.4a}$$

$$\psi_1(r) = \tilde{\psi}_1(r) + \tilde{\psi}_0(r), \qquad \psi_{N-1}(r) = \tilde{\psi}_{N-1}(r) + \tilde{\psi}_N(r). \tag{2.5.4b}$$

After this, we can rewrite Eqs. (2.4.1)

$$\tilde{\psi}_{k-1}(r) + \tilde{\psi}_k(r) + \tilde{\psi}_{k+1}(r) + \tilde{\psi}_{k+2}(r) = 1, \quad r_k < r < r_{k+1}, \quad k = 1, 2, ..., N-2. \tag{2.5.5a}$$

Figure 2.6 shows the situation in the interval $r_k < r < r_{k+1}$. We also let $\tilde{\psi}_0(r_1) = \tilde{\psi}_N(r_{N-1}) = 0$ so that Eq. (2.1.6) also holds for $\tilde{\psi}_k$, i.e., $\tilde{\psi}_k(r_j) = \delta_{jk}$. Furthermore, we let

$$\tilde{\psi}_1'(r_1) = \tilde{\psi}_{N-1}'(r_{N-1}) = 0, \tag{2.5.6}$$

as $\psi_k'(r_k) = 0$ for $k = 2, 3, ..., N-2$ used in Section 2.4. In this way, we obtain from Eqs. (2.4.3)

$$\tilde{\psi}_{k-1}'(r_k) + \tilde{\psi}_{k+1}'(r_k) = 0, \qquad k = 1, 2, ..., N-1, \tag{2.5.7}$$

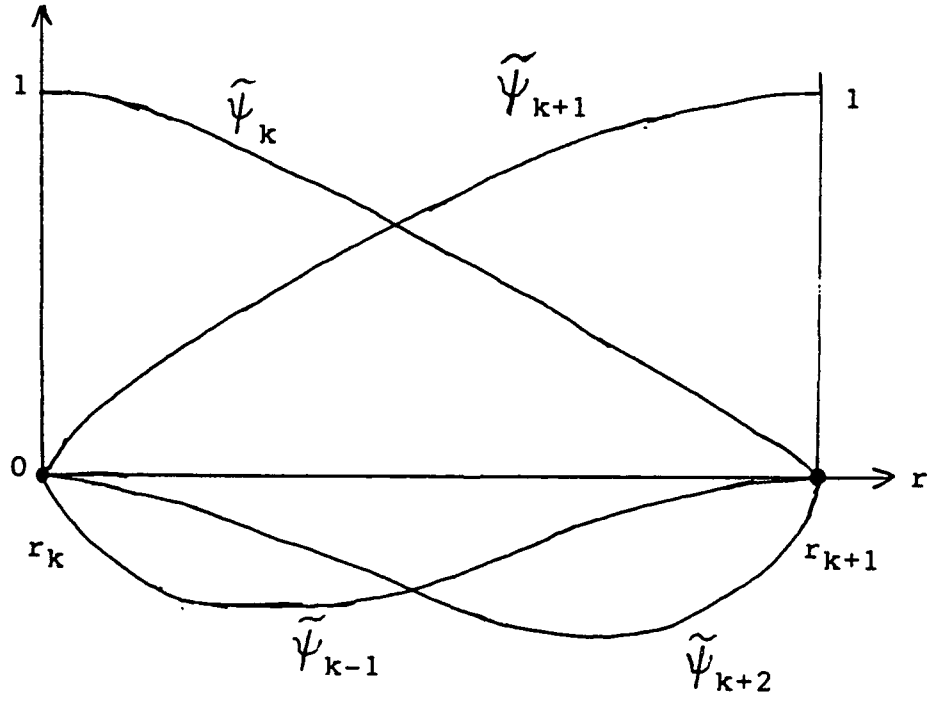



**Figure 2.6.** Illustration for 4 non-zero interpolants in the interval $r_k < r < r_{k+1}$, $k = 1, 2, ..., N-2$.

Corresponding to our choice (2.5.2), we require, within the interval $r_k < r < r_{k+1}$, the 4 areas under the 4 nonzero interpolants to be determined by

$$\int_{r_k}^{r_{k+1}} \tilde{\psi}_{k-1}(r)dr = \eta \tilde{C}_{k-1}, \tag{2.5.8a}$$

$$\int_{r_k}^{r_{k+1}} \tilde{\psi}_k(r)dr = \tilde{C}_k, \tag{2.5.8b}$$

$$\int_{r_k}^{r_{k+1}} \tilde{\psi}_{k+1}(r)dr = \tilde{C}_{k+1}, \tag{2.5.8c}$$

$$\int_{r_k}^{r_{k+1}} \tilde{\psi}_{k+2}(r)dr = \eta\tilde{C}_{k+2} \tag{2.5.8d}$$

for $k = 1, 2, ..., N-2$. It is easy to see that, by summing over both sides of Eqs. (2.5.8) and using Eqs. (2.5.2) and (2.4.1), we get an identity $r_{k+1} - r_k = h_k$, which is just Eq. (2.1.13). We emphasize that the introduction of all tilted quantities is to eliminate the special situation associated with the two end points $r_1$ and $r_{N-1}$ [see Eqs. (2.4.3)] and the two end intervals $r_1 < r < r_2$ and $r_{N-2} < r < r_{N-1}$ [see Eqs. (2.4.1)]. It is easy to see that $\eta = 0$ is a special case, in which the $C^2$-continuity blending functions reduce to the $C^1$-continuity blending functions discussed in Section 2.3.

The piecewise linear and continuous $\psi'_k(r)$ used in Ref. [2] cannot be used here, since there would be 5 parameters one can choose but there are 6 conditions [three from Eqs. (2.5.12) and the other three will be mentioned below]. In our construction, we eliminate the intermediate point $w_k$ and thus reduce the number of regions to be considered. We use piecewise cubic polynomials for the first order derivative of the interpolation functions $\psi_k(r)$,

$$\tilde{\psi}'_k(r) = A_k x_k + B_k x_k^2, \tag{2.5.9a}$$

$$\tilde{\psi}'_{k-1}(r) = -Z_k + A_k^- x_k + B_k^- x_k^2 + (Z_k - A_k^- - B_k^-)x_k^3, \tag{2.5.9b}$$

$$\tilde{\psi}'_{k+1}(r) = \quad Z_k + A_k^+ x_k + B_k^+ x_k^2 - (Z_k + A_k^+ + B_k^+)x_k^3, \tag{2.5.9c}$$

for $k = 1, 2, ..., N-2$. The fourth one $\psi'_{k+2}(r)$ is determined by Eq. (2.4.2) and satisfies $\psi'_{k+2}(r_k) = 0$. There are 6 unknowns for a given interval: three $A$'s and three $B$'s. In Eqs. (2.5.9), using Eq. (2.4.3c), we have defined

$$Z_k \equiv \tilde{\psi}'_{k+1}(r_k) = -\tilde{\psi}'_{k-1}(r_k), \qquad k = 1, 2, ..., N-1, \tag{2.5.10}$$

which are positive, $Z_k > 0$. Carrying out integration once and using Eqs. (2.1.6), we obtain

$$\tilde{\psi}_k(r) = 1 + h_k(\frac{1}{2}A_k x_k^2 + \frac{1}{3}B_k x_k^3), \tag{2.5.11a}$$

$$\tilde{\psi}_{k-1}(r) = h_k[-Z_k x_k + \frac{1}{2}A_k^- x_k^2 + \frac{1}{3}B_k^- x_k^3 - \frac{1}{4}(A_k^- + B_k^- - Z_k)x_k^4], \tag{2.5.11b}$$

$$\tilde{\psi}_{k+1}(r) = h_k[\ Z_k x_k + \frac{1}{2}A_k^+ x_k^2 + \frac{1}{3}B_k^+ x_k^2 - \frac{1}{4}(A_k^+ + B_k^+ + Z_k)x_k^4], \tag{2.5.11c}$$

# THIS PAGE IS MISSING IN ORIGINAL DOCUMENT

22, 23, 24

# CHAPTER III. CONTROL POINT FORM OF ALGEBRAIC GRID GENERATION WITH CLUSTERING

The multisurface transformation reported in Chapter II can be used to generate grids algebraically, which is called the control point form (CPF) of algebraic grid generation [3]. In this chapter, we report our progress made in this area. This chapter is organized as follows. In Section 3.1, we identify the problem with an implementation of the control point form of algebraic grid generation. In Section 3.2, we discuss how to generate a linear-increment straight line when all control points $\mathbf{P}_k$ are given on a straight line. In Section 3.3 we discuss, when applying the multisurface transformation to a 2D grid generation within a parallelogram, how to make the 2D transformation bilinear. In steady of a parallelogram, in Section 3.4 we discuss how to get a bilinear transformation within a quadrilateral. In Section 3.5 we discuss how to deal with general case of 2D grid with clusterings.

## 3.1. Control Point Form of Algebraic Grid Generation

Grid generation in a two-dimensional space can be stated as finding the relation $\mathbf{P}(\xi,\eta)$ between the coordinates $\mathbf{P}=(x,y)$ in the 2D physical space and the corresponding parametric values $(\xi,\eta)$ in a rectangle $\xi_{\min}\le\xi\le\xi_{\max}$, $\eta_{\min}\le\eta\le\eta_{\max}$.

When the multisurface transformation is used for algebraic grid generation in a 2D physical space, an array of control points $\mathbf{P}_k$ is replaced by a net of control points $\mathbf{Q}_{ij}$ ($i=1,2,...,I$ and $j=1,2,...,J$). Let the blending functions in Eq. (2.1.15) be $\alpha_i(\xi)$ ($i=1,2,...,I$) for the $\xi$ direction and $\beta_j(\eta)$ ($j=1,2,...,J$) for the $\eta$ direction (in a parametric space). Let the boundary specification be given by

$$\mathbf{P}(\xi,\eta_1)=\mathbf{u}^1(\xi),\qquad \mathbf{P}(\xi,\eta_{J-1})=\mathbf{u}^2(\xi),$$

$$\mathbf{P}(\xi_1,\eta)=\mathbf{v}^1(\eta),\qquad \mathbf{P}(\xi_{I-1},\eta)=\mathbf{v}^2(\eta). \tag{3.1.1}$$

Starting from the transfinite interpolation (also called Coons patches or the Boolean sum), it has been found that the control point form [3] of algebraic grid generation can produce a 2D grid in the following equivalent way: First, we construct a tensor product,

$$\mathbf{T}(\xi,\eta)=\sum_{i=1}^{I}\sum_{j=1}^{J}\alpha_i(\xi)\beta_j(\eta)\mathbf{Q}_{ij}. \tag{3.1.2}$$

Then, we add edge adjustments for all four boundary curves,

$$\begin{aligned}\mathbf{P}(\xi,\eta)=\mathbf{T}(\xi,\eta)&+\alpha_1(\xi)[\mathbf{v}^1(\eta)-\mathbf{T}(\xi_1,\eta)]+\alpha_I(\xi)[\mathbf{v}^2(\eta)-\mathbf{T}(\xi_{I-1},\eta)]\\&+\beta_1(\eta)[\mathbf{u}^1(\xi)-\mathbf{T}(\xi,\eta_1)]+\beta_J(\eta)[\mathbf{u}^2(\xi)-\mathbf{T}(\xi,\eta_{J-1})].\end{aligned} \tag{3.1.3}$$

The way of assigning the coordinates of grid points in the parametric space (computational domain) is very import for regenerating a given grid using the CPF method. In particular, in the presence of clustering, care must be taken. Since a grid point carries index $(i,j)$, it is uaually assume that the coordinate in the parmetric space is simply

$(\xi, \eta) \propto (i, j)$. In other words, the mesh in the parametric space is a mesh with equal spacing. (Sometimes, the computational domain is normalized to a unit square so that the spacing in the $i$ (or $\xi$) direction is different from that in the $j$ (or $\eta$) direction. Figures 3.1–3.3 show three examples of regenerating 2D grids, including given grids, the mesh distributions in the computational domain, and the regenerated grids using Eq. (3.1.3). We see that even in the simple case of a quadrilateral (Fig. 3.1), the given grid is not reproduced. In the following sections, we show how to overcome this problem.

### 3.2. Generation of a Linear-Increment Straight Line

As a preparation for later sections of this chapter, we study in this section how to generate a straight line $\mathbf{P}(r)$ that increases linearly with the parameter $r$. In other words, we want to know how to choose the control points $\mathbf{P}_k$ and the blending functions $G_k(r)$ so that the curve $\mathbf{P}(r)$ in Eq. (2.1.1) behaves like a straight line and is also a linear function of the parameter $r$,

$$\mathbf{P}(r) = \mathbf{P}_1 + (r - r_1)\mathbf{n}, \tag{3.2.1}$$

where $\mathbf{n}$ denotes the direction of the straight line. It is easy to see that all control points must be on the same straight line,

$$\mathbf{P}_k = \mathbf{P}_1 + (p_k - p_1)\mathbf{n}, \quad k = 2, 3, ..., N. \tag{3.2.2}$$

If the values of $p_k$'s are not given, we can, say, (i) use "half one one ... one half" spacing rule in physical space to determine $p_k$'s and also in parametric space to determine $(C_k, b_k)$ and (ii) use "$h_1 = h_2 = ... = h_{N-2}$" to determine $C^1$ blending functions $G_k(r)$ reported in Sections 2.2 and 2.3. If, on the other hand, we are given the values of $p_k$, then the question becomes how to choose $b_k$. We find that, if we choose

$$b_k = p_k, \qquad k = 1, 2, ..., N. \tag{3.2.3}$$

then, using Eqs. (2.1.1), (2.1.7) and (2.1.8), we obtain the linear relation (3.2.1).

Using the linear-increment straight line (3.2.1), for any given set of (curve) points

$$\mathbf{c}_i = \mathbf{P}_1 + c_i\mathbf{n}, \qquad i = 1, 2, 3, ... \tag{3.2.4}$$

(not control points), we can reproduce these points $\mathbf{c}_i$ exactly by letting their parametric values to be $c_i + r_1$. If we use a set of control points specified by Eq. (3.2.2) in the physical space and by Eq. (3.2.3) in the parametric space, we can also regenerate (curve) points $\mathbf{c}_i$ exactly using the multisurface transformation (2.1.1). The advantage of using Eq. (2.1.1) is that, by reproducing $\mathbf{c}_i$ initially, we can subsequently move one or more control points $\mathbf{P}_k$ and thus modify the positions of (curve) points $\mathbf{c}_i$, $i = 1, 2, 3, ...$ .

### 3.3. Bilinear Transformation for Grid Generation in a Parallelogram

In this section, we consider the case in which the physical region where a 2D grid is to be built is a parallelogram, i.e., the physical region is a four-sided 2D area whose opposite sides are parallel and equal. In such a case, the net of control points can be specified easily

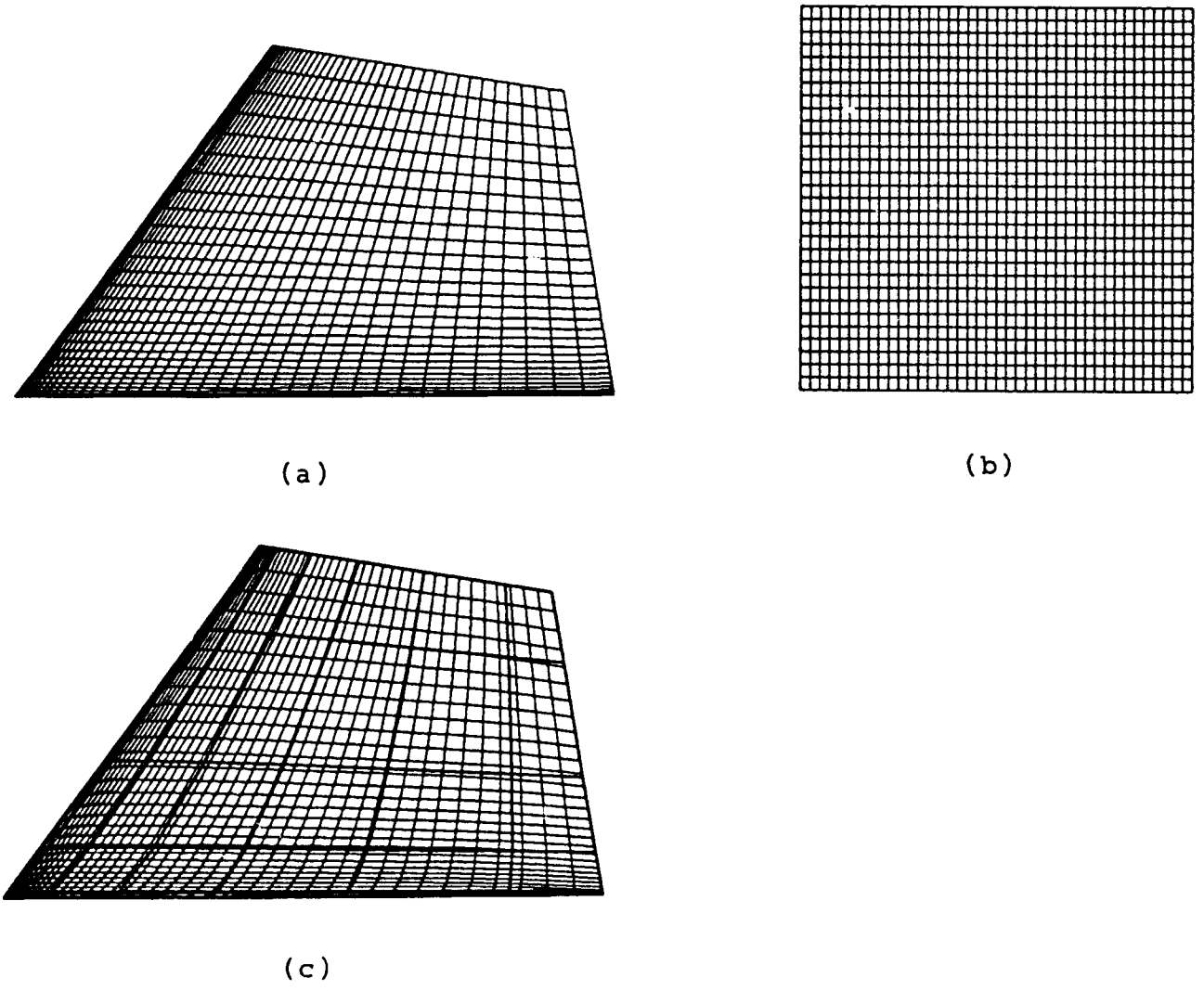
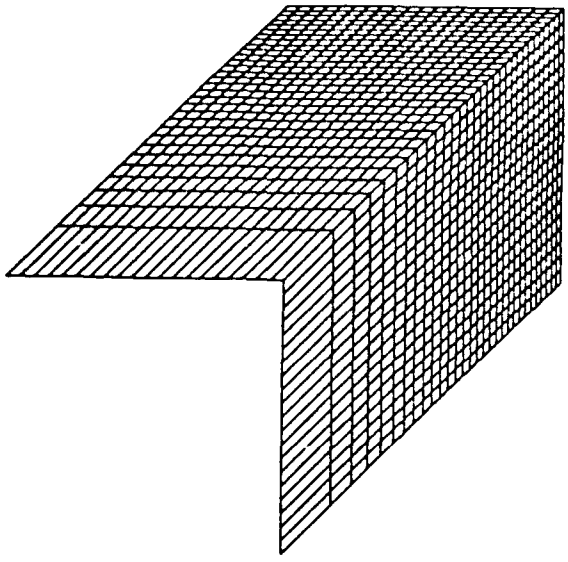
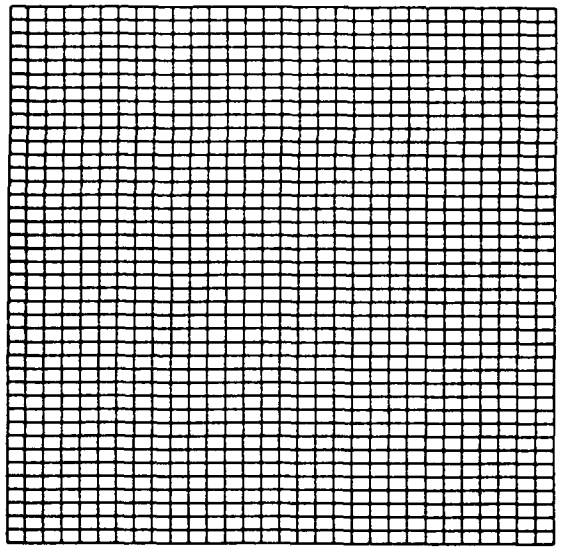


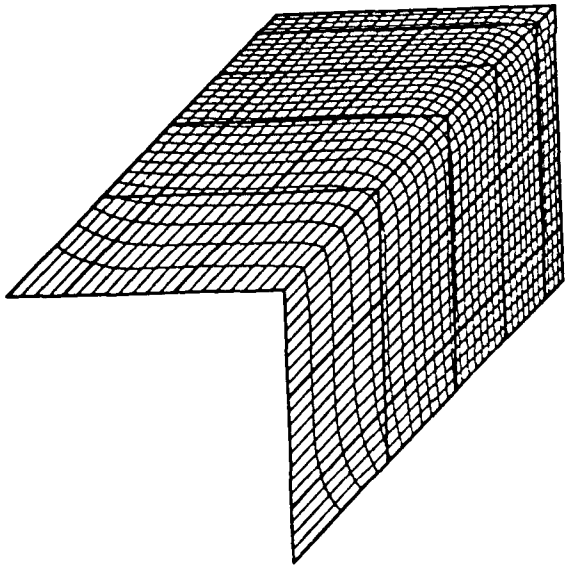

**Figure 3.1.** (a) A given grid with clustering along two edges within a quadrilateral. (b) The uniform distribution of grid points in the computational domain (i.e., a parametric space) used in the CPF. (c) The grid regenerated using the control point form of algebraic grid generation and the uniform grid distribution in the computational domain shown in (b). The sparse and thick net is the net of control points, and its distribution in the parametric space is of the "half one one ... one half" type given in Eq. (2.3.17).



(a)



(b)



(c)

**Figure 3.2.** (a) An initial grid with clustering along one edge. (b) The uniform distribution of grid points in the computational domain used in the CPF. (c) The grid regenerated using the CPF method and the uniform grid distribution in the computational domain shown in (b). The sparse and thick net is the control net, and its distribution in the parametric space is of the "half one one ... one half" type given in Eq. (2.3.17).

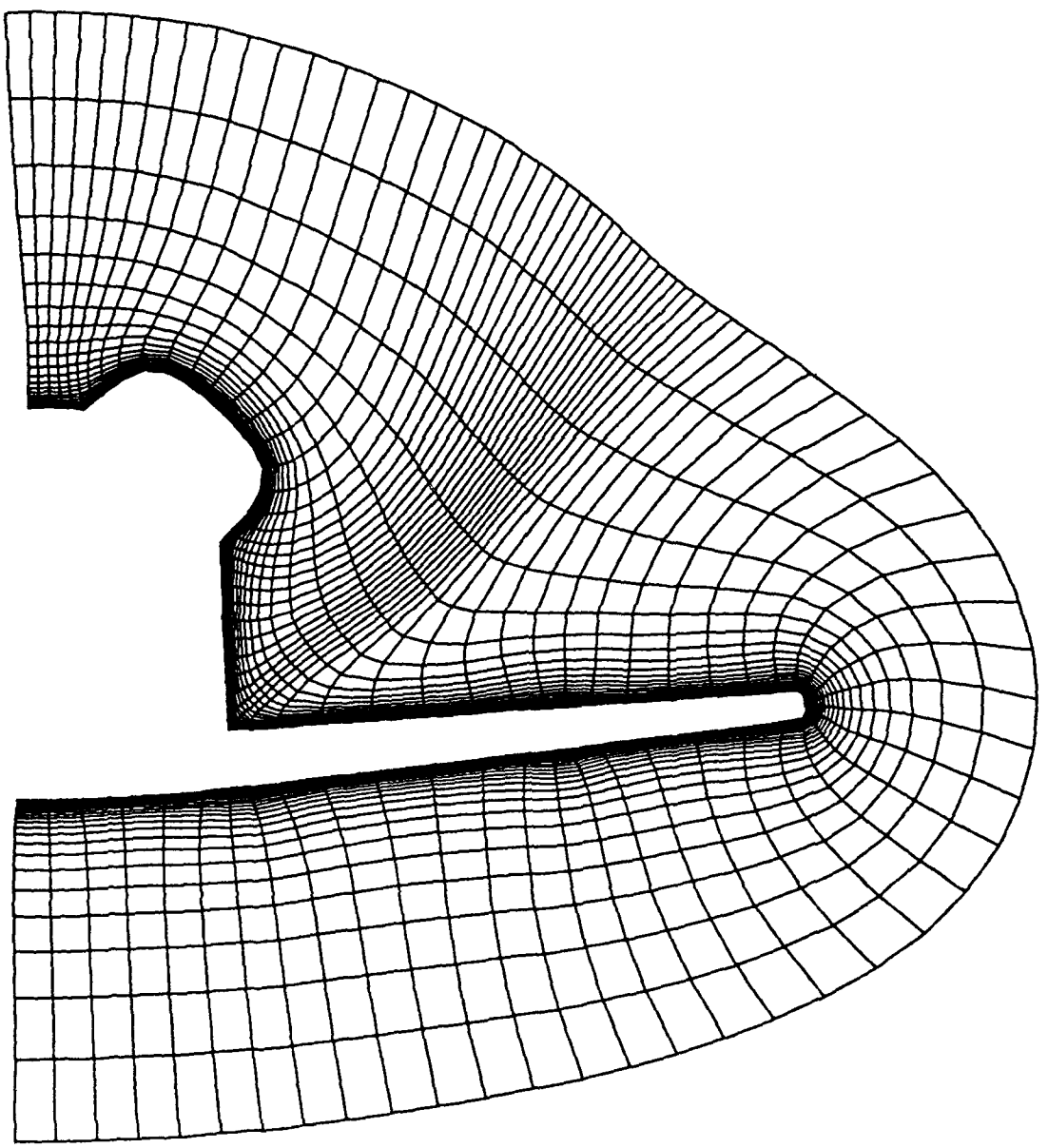


**Figure 3.3.** (a) A sheet of space shuttle grid (near the tail of the space shuttle, only half grid is shown), which has clustering along the body of the space shuttle.

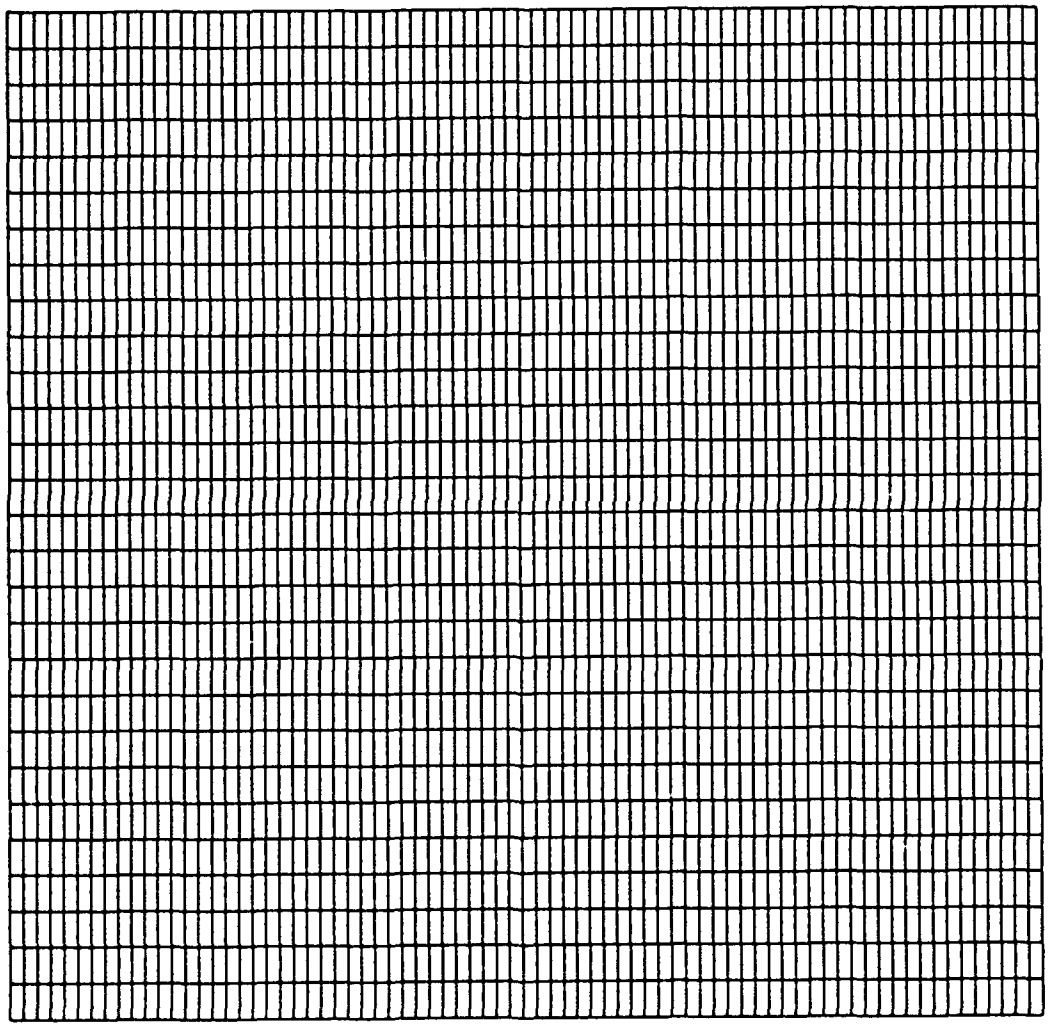


**Figure 3.3.** (b) The uniform distribution of grid points in the parametric space used in the CPF.

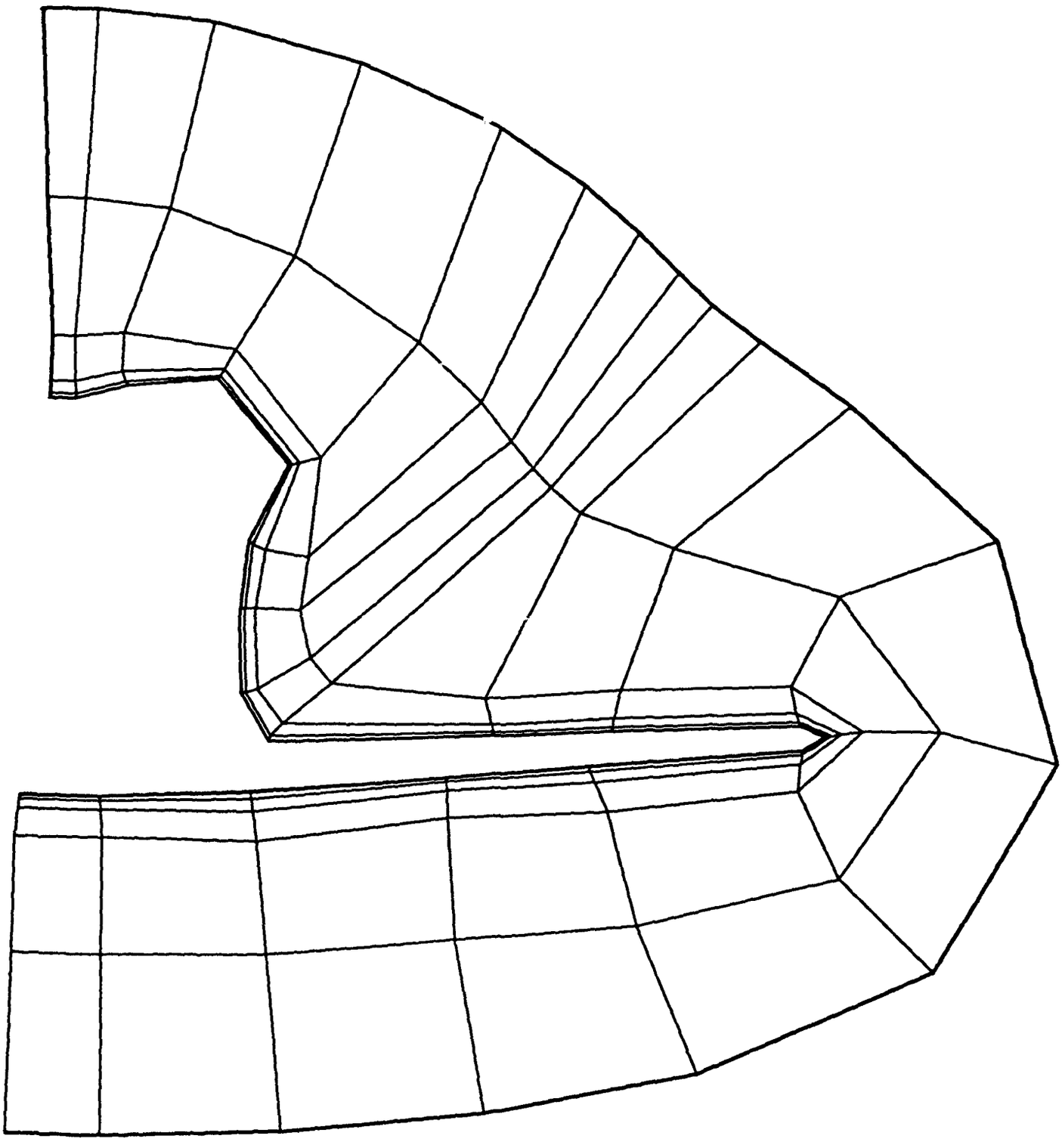


**Figure 3.3.** (c) The control net attached to the 2D grid given in (a). Its distribution in the parametric space is of the "half one one ... one half" type given in Eq. (2.3.17).

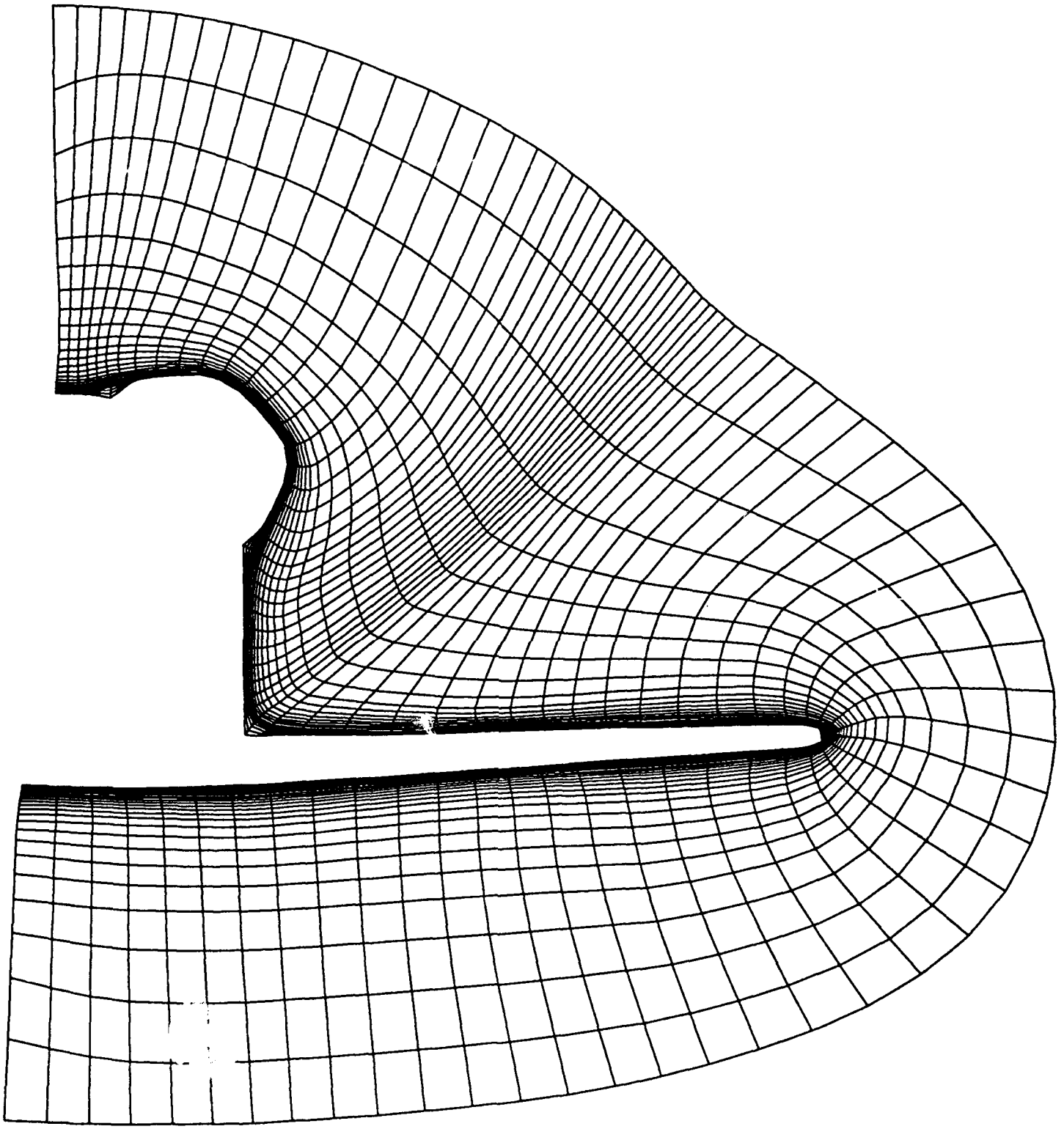


**Figure 3.3.** (d) The space shuttle grid regenerated using the CPF method and the uniform grid distribution in the parametric space shown in (b).

$$\mathbf{Q}_{ij} = \mathbf{P}(\xi_1, \eta_1) + (\xi_i - \xi_1)\mathbf{e}_1 + (\eta_j - \eta_1)\mathbf{e}_2, \tag{3.3.1}$$

where $\mathbf{e}_1$ and $\mathbf{e}_2$ give the two (non-parallel in general) directions (non-orthogonal in general) of the parallelogram, and $(\xi_i, \eta_j)$ is the coordinate of the control point $\mathbf{Q}_{ij}$ in the parametric space. Now, similar to the linear relation given in Eq. (3.2.1), we want the 2D transformation $\mathbf{P}(\xi, \eta)$ to be bilinear in $\xi$ and $\eta$,

$$\mathbf{P}(\xi, \eta) = \mathbf{P}(\xi_1, \eta_1) + (\xi - \xi_1)\mathbf{e}_1 + (\eta - \eta_1)\mathbf{e}_2. \tag{3.3.2}$$

Let us see how we can get Eq. (3.3.2). First, we notice from Eq. (2.1.17) that, as a general rule of surface generation, we have

$$\sum_{i=1}^{I} \alpha_i(\xi) = 1, \qquad \sum_{j=1}^{J} \beta_j(\eta) = 1. \tag{3.3.3}$$

Then, from the uniformity condition (2.1.18), we have

$$\sum_{i=1}^{I} \alpha_i(\xi) b_i = \xi, \qquad \sum_{j=1}^{J} \beta_j(\eta) b'_j = \eta \tag{3.3.4}$$

in the 2D case. Substituting Eq. (3.3.1) into Eq. (3.1.2) and using Eqs. (3.3.3) , we obtain

$$\mathbf{T}(\xi, \eta) = \mathbf{P}(\xi_1, \eta_1) + \sum_{i=1}^{I} \alpha_i(\xi)\xi_i \mathbf{e}_1 - \xi_1 \mathbf{e}_1 + \sum_{j=1}^{J} \beta_j(\eta)\eta_j \mathbf{e}_2 - \eta_1 \mathbf{e}_2. \tag{3.3.5}$$

With the choice $b_i = \xi_i$ and $b'_j = \eta_j$ for constructing the blending functions and using Eq. (3.3.4), we simplify Eq. (3.3.5) to

$$\mathbf{T}(\xi, \eta) = \mathbf{P}(\xi_1, \eta_1) + (\xi - \xi_1)\mathbf{e}_1 + (\eta - \eta_1)\mathbf{e}_2. \tag{3.3.6}$$

The bilinear relation (3.3.2) can be obtained either by letting $\mathbf{P}(\xi, \eta) = \mathbf{T}(\xi, \eta)$ directly [i.e., without edge boundary adjustment terms in (3.1.3)] or by parametizing the four boundary lines according to the linear increment rule:

$$\begin{aligned} \mathbf{u}^1(\xi) &= \mathbf{P}(\xi_1, \eta_1) + (\xi - \xi_1)\mathbf{e}_1, \quad \mathbf{u}^2(\xi) = \mathbf{P}(\xi_1, \eta_{J-1}) + (\xi - \xi_1)\mathbf{e}_1, \\ \mathbf{v}^1(\eta) &= \mathbf{P}(\xi_1, \eta_1) + (\eta - \eta_1)\mathbf{e}_2, \quad \mathbf{v}^2(\eta) = \mathbf{P}(\xi_{I-1}, \eta_1) + (\eta - \eta_1)\mathbf{e}_2. \end{aligned} \tag{3.3.7}$$

With the CPF constructed in this way, we have a bilinear CPF transformation within a parallelogram. Consequently, for any given grid within a parallelogram, we can regenerate the given grid exactly provided that we determine the coordinate $(\xi, \eta)$ of each grid (control) point in the parametric space using Eq. (3.3.2) [Eq. (3.3.1)]. Afterwards, we can move one or more control points $\mathbf{Q}_{ij}$ to manipulate the 2D grid.

### 3.4. Bilinear Transformation for Grid Generation in a Quadrilateral

In this section, we consider 2D grid generation within a quadrilateral, i.e., the physical region is a four-sided 2D area whose boundaries are 4 straight lines. In such a case, we first have to normalize the 2D parametric space to a unit square, $0 = \xi_1 \le \xi \le \xi_{I-1} = 1$ and $0 = \eta_1 \le \eta \le \eta_{J-1} = 1$. The net of control points should be established now according to

$$\begin{aligned}\mathbf{Q}_{ij} = &(1-\xi_i)(1-\eta_j)\mathbf{P}(\xi_1,\eta_1) + \xi_i(1-\eta_j)\mathbf{P}(\xi_{I-1},\eta_1) \\ &+ (1-\xi_i)\eta_j\mathbf{P}(\xi_1,\eta_{J-1}) + \xi_i\eta_j\mathbf{P}(\xi_{I-1},\eta_{J-1}),\end{aligned} \tag{3.4.1}$$

where $(\xi_i,\eta_j)$ is the coordinate of the control point $\mathbf{Q}_{ij}$ within the unit square. Now, instead of the bilinear relation in Eq. (3.3.2), we want the 2D transformation $\mathbf{P}(\xi,\eta)$ to be in a different bilinear form,

$$\mathbf{P}(\xi,\eta) = (1-\xi)(1-\eta)\mathbf{P}(\xi_1,\eta_1)+\xi(1-\eta)\mathbf{P}(\xi_{I-1},\eta_1)+(1-\xi)\eta\mathbf{P}(\xi_1,\eta_{J-1})+\xi\eta\mathbf{P}(\xi_{I-1},\eta_{J-1}); \tag{3.4.2}$$

i.e., a bilinear relation based on 4 corner points. Let us see how we can get Eq. (3.4.2). Substituting Eq. (3.4.1) into Eq. (3.1.2), we obtain

$$\begin{aligned}\mathbf{T}(\xi,\eta) = \quad &\mathbf{P}(\xi_1,\eta_1)\left[\sum_{i=1}^{I}\alpha_i(\xi)(1-\xi_i)\right]\sum_{j=1}^{J}\beta_j(\eta)(1-\eta_j) \\ &+ \mathbf{P}(\xi_{I-1},\eta_1)\left[\sum_{i=1}^{I}\alpha_i(\xi)\xi_i\right]\sum_{j=1}^{J}\beta_j(\eta)(1-\eta_j) \\ &+ \mathbf{P}(\xi_1,\eta_{J-1})\left[\sum_{i=1}^{I}\alpha_i(\xi)(1-\xi_i)\right]\sum_{j=1}^{J}\beta_j(\eta)\eta_j \\ &+ \mathbf{P}(\xi_{I-1},\eta_{J-1})\left[\sum_{i=1}^{I}\alpha_i(\xi)\xi_i\right]\sum_{j=1}^{J}\beta_j(\eta)\eta_j.\end{aligned} \tag{3.4.3}$$

With the choice $b_i = \xi_i$ and $b'_j = \eta_j$ for constructing the blending functions and using Eqs. (3.3.3) and (3.3.4), we arrive at

$$\mathbf{T}(\xi,\eta) = (1-\xi)(1-\eta)\mathbf{P}(\xi_1,\eta_1)+\xi(1-\eta)\mathbf{P}(\xi_{I-1},\eta_1)+(1-\xi)\eta\mathbf{P}(\xi_1,\eta_{J-1})+\xi\eta\mathbf{P}(\xi_{I-1},\eta_{J-1}). \tag{3.4.4}$$

Similar to the situation in Section 3.3, the bilinear relation (3.4.2) can be obtained either by letting $\mathbf{P}(\xi,\eta) = \mathbf{T}(\xi,\eta)$ directly or by parametizing the four boundary lines as

$$\begin{aligned}&\mathbf{u}^1(\xi) = (1-\xi)\mathbf{P}(\xi_1,\eta_1) + \xi\mathbf{P}(\xi_{I-1},\eta_1), \quad \mathbf{u}^2(\xi) = (1-\xi)\mathbf{P}(\xi_1,\eta_{J-1}) + \xi\mathbf{P}(\xi_{I-1},\eta_{J-1}), \\ &\mathbf{v}^1(\eta) = (1-\eta)\mathbf{P}(\xi_1,\eta_1) + \eta\mathbf{P}(\xi_1,\eta_{J-1}), \quad \mathbf{v}^2(\eta) = (1-\eta)\mathbf{P}(\xi_{I-1},\eta_1) + \eta\mathbf{P}(\xi_{I-1},\eta_{J-1}).\end{aligned} \tag{3.4.5}$$

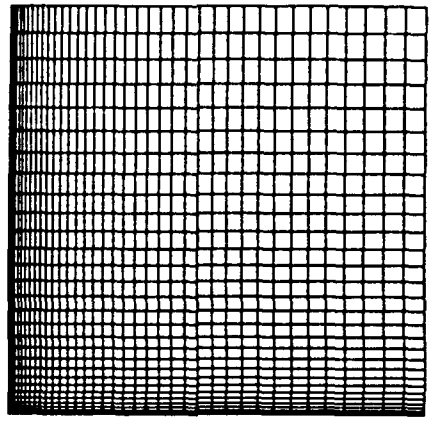
Having a 2D CPF relation built in this scheme, we get a bilinear CPF transformation within a quadrilateral. Thus, for any given 2D grid within a quadrilateral, we can regenerate the given grid exactly provided that we solve the coordinate $(\xi, \eta)$ of each grid (control) point in the parametric space using Eq. (3.4.2) [Eq. (3.4.1)]. The grid manipulation can be achieved by moving one or more control points $\mathbf{Q}_{ij}$ at a user's discretion. Note that the results in this section can be regard as a generalization of the results in Section 3.3, since a parallelogram is in fact a special quadrilateral.

For a 2D grid with clustering either within a parallelogram or within a quadrilateral, it is easy to see from either Eq. (3.3.2) or Eq. (3.4.2) that the image of the grid in the parametric space $(\xi, \eta)$ is also a grid with clustering. This result shows that, for a grid with clustering, its image in the parametric space $(\xi, \eta)$ should not be with equal spacings; i.e., the mesh in the parametric space should not be like those displayed in Figs. 3.1(b), 3.2(b), and 3.3(b). In other words, we should use three spaces in the control point form of algebraic grid generation. In stead of the usual two spaces [physical space $(x, y)$ and curvilinear space $(\xi, \eta) \propto (i, j)$], we should use the following three spaces: the physical region $(x, y)$ (arbitrary shape), a parametric space $(\xi, \eta)$ (a rectangle), and an index space $(i, j)$ (also a rectangle). In the index space $(i, j)$, the grid points are always uniformly distributed and the grid lines are always straight lines. In the parametric space $(\xi, \eta)$, depending on the situation in the physical region, the grid points can be either uniformly or non-uniformly distributed and the grid lines may or may not be straight lines. In Fig. 3.4 we show how to regenerate exactly the grid given in Fig. 3.1(a), which is within a quadrilateral.

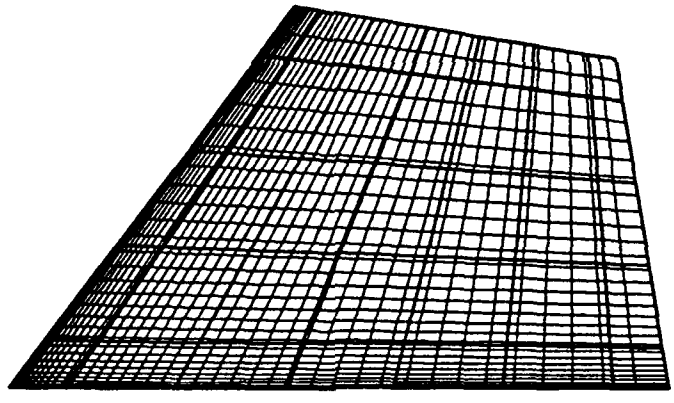
### 3.5. General 2D Physical Regions

The conclusion reached at the end of Section 3.4 is valid not only when the physical region is a quadrilateral (it contains the parallelogram as a special case) but also for a physical region of any shape; Namely, we need three spaces for the CPF to regenerate a given grid.

For a given 2D grid built within a general 2D region, we cannot have a bilinear transformation like those in Eqs. (3.3.2) and (3.4.2). In order to regenerate the given grid using the control point form of algebraic grid generation, we should first reproduce the clustering patterns of the given grid in the parametric space $(\xi, \eta)$ using, say, arc length measurements. In general, such an arc-length method will not reproduce a given grid exactly. It will, however, capture the general clustering tendency of the given grid. Figure 3.5 shows how to regenerate the grid given in Fig. 3.2(a). In the region of 90-degree turn, the CPF grid does not match the given grid exactly; Otherwise, the CPF grid matches the given grid exactly. Figure 3.6 shows how to regenerate the grid given in Fig. 3.3(a) (a sheet of grid for the space shuttle). Comparing Fig. 3.6(b) with Fig. 3.3(d), we see the improvements near the convex regions along the edge with clustering (i.e., $\eta = \eta_{\min}$ edge). To reproduce an arbitrarily given grid exactly, we have to subsequently search and adjust the parametric coordinate $(\xi, \eta)$ of each grid point so that the gird lines in the parametric space are not straight lines in general.

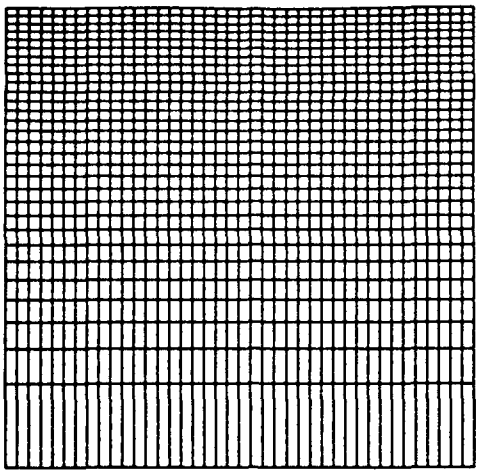


(a)

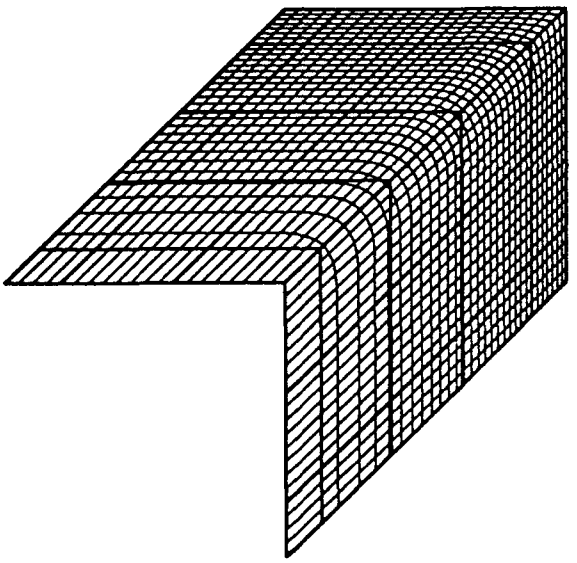


(b)

**Figure 3.4.** Regenerating the grid of Fig. 3.1(a) using an improved implementation of the CPF method. (a) The non-uniform grid distribution in the parametric space $(\xi, \eta)$ determined from Eq. (3.4.2). (b) The grid regenerated using the control point form of algebraic grid generation and the non-uniform grid distribution in the parametric space shown in (a). The sparse and thick net is the control net, and its distribution in the parametric space is of the "half one one ... one half" type given in Eq. (2.3.17).



(a)



(b)

**Figure 3.5.** Regenerating the grid of Fig. 3.2(a) using an improved implementation of the CPF method. (a) The non-uniform grid distribution in the parametric space $(\xi, \eta)$ determined according to the grid spacing in the physical space. (b) The CPF grid generated using the non-uniform grid distribution in the parametric space shown in (a). The sparse and thick net is the control net, and its distribution in the parametric space is of the "half one one ... one half" type given in Eq. (2.3.17).

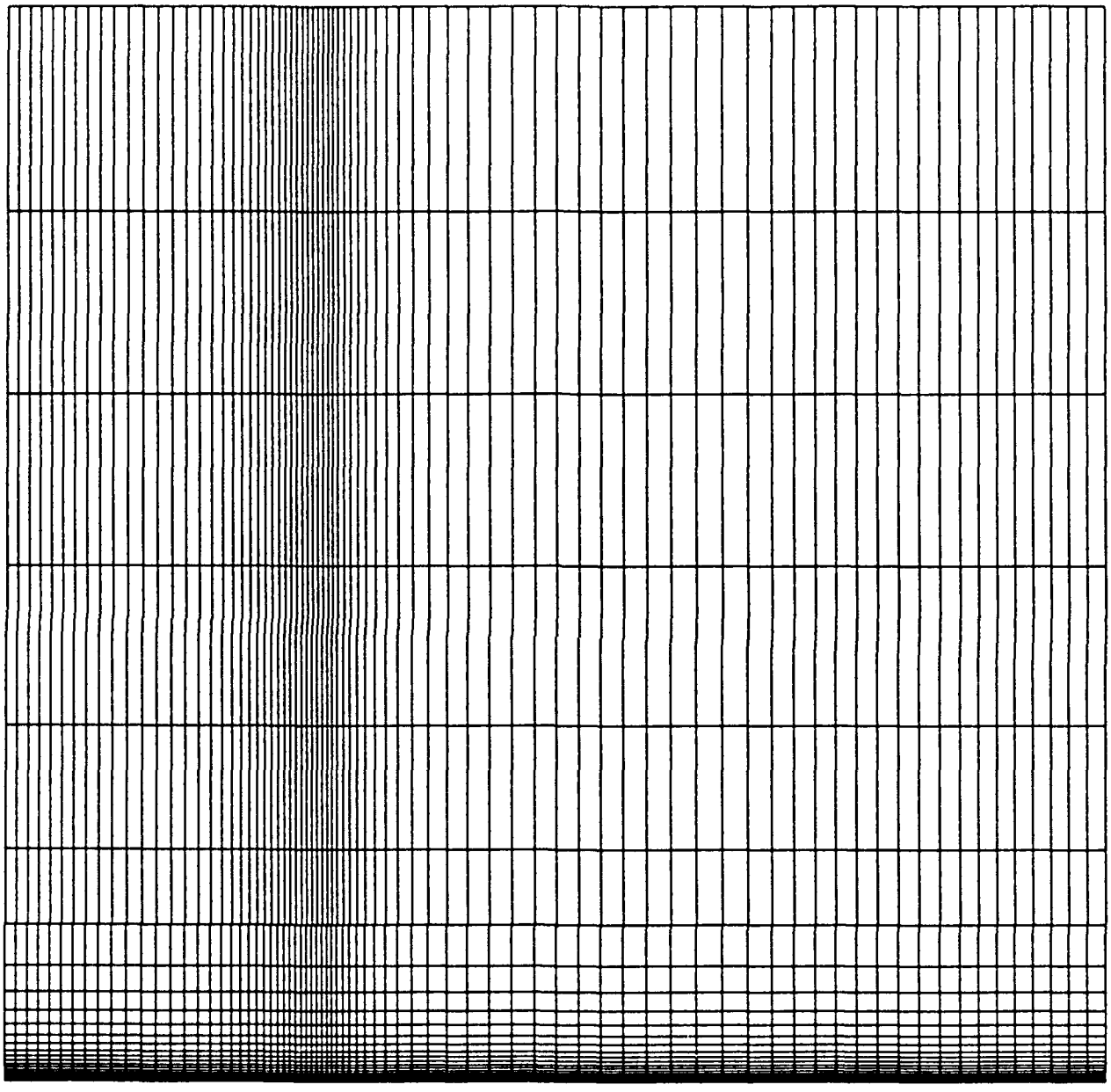


**Figure 3.6.** Regenerating the grid of Fig. 3.3(a) using an improved implementation of the CPF method. (a) The non-uniform grid distribution in the parametric space $(\xi, \eta)$ obtained through an arc-length measurement of the space shuttle grid in the physical space.

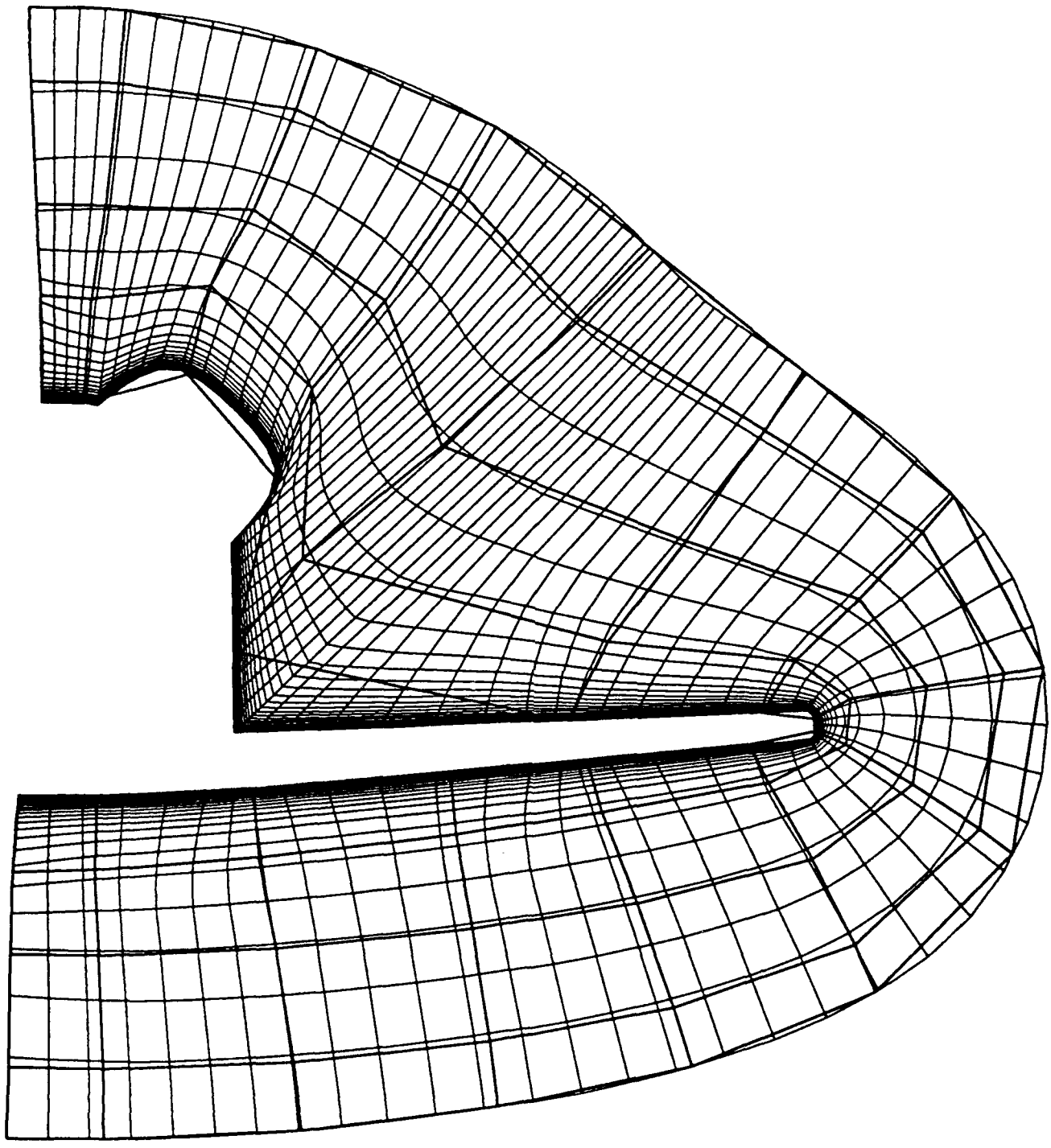


**Figure 3.6.** (b) The space shuttle grid regenerated using the CPF method and the non-uniform grid distribution in the parametric space shown in (a). The sparse and thick net is the control net, and its distribution in the parametric space is of the "half one one ... one half" type given in Eq. (2.3.17).

# CHAPTER IV CURVATURE CORRECTION

## 4.1 Basic Concepts of Elliptic Grid Generation

The underlying idea behind all grid generation methods is to express the physical coordinates of a point in space, in terms of a set of variables in the computational domain. For each set of physical coordinates (x,y,z), there is a corresponding set of points ($\xi,\eta,\zeta$) in the transformed or computational domain. The mathematical expression of the above statement is:

$$x = x(\xi,\eta,\zeta) \Leftrightarrow \xi = \xi(x,y,z)$$
$$y = y(\xi,\eta,\zeta) \Leftrightarrow \eta = \eta(x,y,z)$$
$$z = z(\xi,\eta,\zeta) \Leftrightarrow \zeta = \zeta(x,y,z)$$

Among the many techniques available for numerical grid generation, elliptic techniques are perhaps the most widely used. We restrict the discussion here to two dimensions. We start by writing a set of elliptic equations for the transformed variables ($\xi,\eta$). The simplest elliptic equation is Laplace's equation:

$$\xi_{xx} + \xi_{yy} = 0 \qquad (4.1.1)$$
$$\eta_{xx} + \eta_{yy} = 0 \qquad (4.1.2)$$

These equations state that given values of the transformed variables $\xi$ and $\eta$ or their derivatives on the boundary of an arbitrary physical domain, we can compute values of $\xi$ and $\eta$ inside the domain. However, our goal is to express x and y inside the physical domain in terms of $\xi$ and $\eta$. Let us assume the computational or transformed domain is rectangular and $\xi$ and $\eta$ are distributed uniformly over this domain. Let the increments $\Delta\xi = \xi_{i+1,j} - \xi_{i-1,j}$ and $\Delta\eta = \eta_{i,j+1} - \eta_{i,j-1}$ be equal to 2 for convenience. The computational or transformed domain is thus a rectangular region with a rectangular mesh covering it. Our goal is to find values for x and y at each point on the computational domain, given values of x and y or their derivatives on the boundaries.

Exchanging the role of the dependent and independent variables in Equations (4.1.1) and (4.1.2) transforms this set of equations to:

$$\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} = 0 \qquad (4.1.3)$$

$$\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} = 0 \tag{4.1.4}$$

where the terms:

$$\alpha = x_\eta^{\,2} + y_\eta^{\,2} \tag{4.1.5}$$

$$\beta = x_\xi x_\eta + y_\xi y_\eta \tag{4.1.6}$$

$$\gamma = x_\xi^{\,2} + y_\xi^{\,2} \tag{4.1.7}$$

are responsible for the coupling between the two equations.

It should be immediately apparent that the transformed equations do indeed represent equations for x and y in terms of $\xi$ and $\eta$.

Equations (4.1.3) and (4.1.4) are discretized and solved simultaneously for x and y. The discrete form of the equations is simple, because the spacing in the computational domain is uniform and the increments are equal to 1. For example, the discretized form of (4.1.3), based on central differences is:

$$\alpha_{i,j} = (\tfrac{1}{2}(x_{i,j+1} - x_{i,j-1}))^2 + (\tfrac{1}{2}(y_{i,j+1} - y_{i,j-1}))^2$$

$$\beta_{i,j} = \tfrac{1}{4}(x_{i+1,j} - x_{i-1,j})(x_{i,j+1} - x_{i,j-1}) + \tfrac{1}{4}(y_{i+1,j} - y_{i-1,j})(y_{i,j+1} - y_{i,j-1})$$

$$\gamma_{i,j} = (\tfrac{1}{2}(x_{i+1,j} - x_{i-1,j}))^2 + (\tfrac{1}{2}(y_{i+1,j} - y_{i-1,j}))^2$$

$$\alpha_{i,j}(x_{i+1,j} - 2x_{i,j} + x_{i-1,j}) + \gamma_{i,j}(x_{i,j+1} - 2x_{i,j} + x_{i,j-1})$$
$$- \tfrac{1}{2}\beta_{i,j}(x_{i+1,j+1} - x_{i-1,j+1} - x_{i+1,j-1} + x_{i-1,j-1}) = 0 \tag{4.1.8a}$$

Similarly for (4.1.4):

$$\alpha_{i,j}(y_{i+1,j} - 2y_{i,j} + y_{i-1,j}) + \gamma_{i,j}(y_{i,j+1} - 2y_{i,j} + y_{i,j-1})$$
$$- \tfrac{1}{2}\beta_{i,j}(y_{i+1,j+1} - y_{i-1,j+1} - y_{i+1,j-1} + y_{i-1,j-1}) = 0 \tag{4.1.8a}$$

## 4.2 Properties of the Elliptic System

Elliptic systems exhibit the desirable tendency to smooth out the grid inside a domain, even if the boundary is not smooth. They also guarantee that the grid lines do not cross. These properties lead to their popularity.

The behavior of elliptic systems near curved boundaries will be the focus of our discussion. Elliptic grids behave differently near concave and convex boundaries. In general, they behave well if the boundary is convex when viewed from the interior of the domain and behave poorly if the boundary is concave. This is because the grid lines tend to cluster near convex regions at the expense of concave regions. This behavior manifests itself in two ways: If the points on the boundaries are fixed, the normal spacing next to concave boundaries can becomes orders of magnitude larger than the spacing next to convex boundaries. If the points are allowed to float along the boundary, in addition to the previous phenomena, the points tend to migrate away from concave corners along the boundary, resulting in poor resolution of the boundary curve.

### 4.3 Curvature Control

The deterioration of grid quality resulting from the behavior of elliptic systems at curved boundaries is severe enough to warrant corrective measures. The most intuitive approach is to deal with the elliptic system on a discrete level and try to find the terms responsible for this behavior. Once the guilty terms have been identified, they can be eliminated or damped, keeping in mind that eliminating the ill effects of these terms near concave corners, also diminishes their favorable effect along convex corners. Our primary goal will be to moderate the effect of curvature on the grid, not to eliminate it.

Obtaining a numerical solution to the pair of elliptic equations (4.1.8a) and (4.1.8b) begins by starting with an initial grid whose point distribution does not satisfy those equations. The points of the initial grid must move to a new position, until they satisfy equations (4.1.8a) and (4.1.8b) and their boundary conditions. We will try to understand the exact nature of the forces that cause the points which do not satisfy the elliptic equations, to move until the grid does satisfy the elliptic equations. We will confine our discussion to grids with orthogonal boundaries, and restrict the regions where we wish to exercise curvature control to the vicinity of the boundaries. For regions of the grid which are orthogonal:

$$\beta = x_\xi x_\eta + y_\xi y_\eta = 0 \qquad (4.3.1)$$

Notice that although an orthogonal grid satisfies equation (4.3.1), simply dropping $\beta$ in equations (4.1.8a) and (4.1.8b) does not force equation (4.3.1) to be satisfied and therefore will not guarantee an orthogonal grid. Assuming that we have, by some means, forced the grid to be

orthogonal in a region, we can then set $\beta = 0$ in that region. This simplifies equations (4.1.8a) and (4.1.8b) considerably:

$$\alpha_{i,j}\,(x_{i+1,j} - 2x_{i,j} + x_{i-1,j}) + \gamma_{i,j}\,(x_{i,j+1} - 2x_{i,j} + x_{i,j-1}) = 0 \qquad (4.3.2a)$$

$$\alpha_{i,j}\,(y_{i+1,j} - 2y_{i,j} + y_{i-1,j}) + \gamma_{i,j}\,(y_{i,j+1} - 2y_{i,j} + y_{i,j-1}) = 0 \qquad (4.3.2b)$$

Solving equation (4.3.2) for $x_{i,j}$ and dropping the i,j subscript from $\alpha_{i,j}$ and $\gamma_{i,j}$:

$$x_{i,j} = \frac{1}{2}\frac{\alpha}{\alpha + \gamma}\,(x_{i+1,j} + x_{i-1,j}) + \frac{1}{2}\frac{\gamma}{\alpha + \gamma}\,(x_{i,j+1} + x_{i,j-1}) \qquad (4.3.3a)$$

A similar expression can be written for $y_{i,j}$:

$$y_{i,j} = \frac{1}{2}\frac{\alpha}{\alpha + \gamma}\,(y_{i+1,j} + y_{i-1,j}) + \frac{1}{2}\frac{\gamma}{\alpha + \gamma}\,(y_{i,j+1} + y_{i,j-1}) \qquad (4.3.3b)$$

For regions of the grid which are orthogonal, the point at (i,j) depends only on its four neighbors at (i+1,j), (i−1,j), (i,j+1) and (i,j−1). As was mentioned earlier, this discrete form of the elliptic equation forms the basis of our attempt to isolate the terms responsible for the undesirable behavior of the elliptic system at concave boundaries.

Typically, for all the points in the field which are in orthogonal regions of the grid, the point at (i,j) is forced to move to a new location, as to satisfy equations (4.3.3a) and (4.3.3b). The tendency of the points to move away from concave corners, is part of this motion. In order to moderate this tendency, we must understand the exact nature of this movement and try to isolate its different components. Denote the position vector at the point at (i,j) by $\vec{P}_{i,j}$:

$$\vec{P}_{i,j} = x_{i,j}\,\hat{i} + y_{i,j}\,\hat{j}$$

It is evident that in order for this point to be in an equilibrium positions relative to its neighbors in an orthogonal region of the grid, $x_{i,j}$ and $y_{i,j}$ must be given respectively by equations (4.3.3a) and (4.3.3b). The relative motion vector $\vec{R}_{i,j}$ is defined as the difference in the position of a point before and after it satisfies equations (4.3.3a) and (4.3.3b). In symbolic form:

$$\vec{R}_{i,j} = (\vec{P}_{i,j})_{new} - (\vec{P}_{i,j})_{old}$$

For a fully converged grid $\vec{R}_{i,j}$ tends to zero every where in the domain. To distinguish between orthogonal and non orthogonal regions, we use the subscript 1 and 2. $\vec{R}_{1_{i,j}}$ satisfies the elliptic equations everywhere, whereas $\vec{R}_{2_{i,j}}$ is restricted to orthogonal regions. We have thus identified the vector which describes the direction and magnitude of the motion of a point. We must now find its different components.

Some manipulation of equations (4.3.3a) and (4.3.3b) shows that $\vec{R}_{2_{i,j}}$ can be expressed as the sum of two vectors which depend only on the difference in the current position of the center point and its four neighboring points. If we let the vector $\vec{V}_1$ contain the contributions from i + 1 and i - 1 and the vector $\vec{V}_2$ contain the contribution from j + 1 and j – 1, the following holds true:

$$\vec{R}_{2_{i,j}} = \vec{V}_1 + \vec{V}_2 \qquad (4.3.4)$$

Figure 4.1 is a graphical representation of the elliptic equations, showing the contribution of the neighboring points to the final location of the center point.

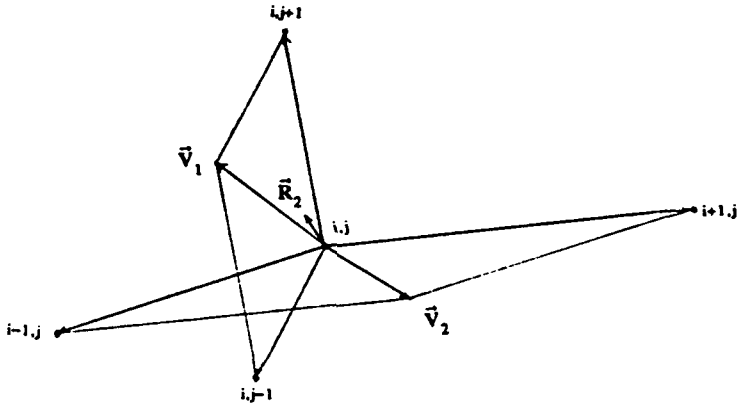



**Figure 4.1**

To derive equation (4.3.4) formally, it is convenient to define two new scalar quantities a and b, such that:

$$a = \frac{\alpha}{\alpha + \gamma} \qquad ; \qquad b = \frac{\gamma}{\alpha + \gamma}$$

using these definitions in (4.3.3a) and (4.3.3b), and subtracting $(x_{i,j})_{old}$ and $(y_{i,j})_{old}$ from both sides of these equations respectively, the Cartesian components of the vectors $\vec{V}_1$ and $\vec{V}_2$ can be expressed as:

$$\vec{V}_1 = \frac{1}{2}a\,(\Delta x_i^+ - \Delta x_i^-)\,\hat{i} + \frac{1}{2}a\,(\Delta y_i^+ - \Delta y_i^-)\,\hat{j} \qquad (4.3.5a)$$

$$\vec{V}_2 = \frac{1}{2}b\,(\Delta x_j^+ - \Delta x_j^-)\,\hat{i} + \frac{1}{2}b\,(\Delta y_j^+ - \Delta y_j^-)\,\hat{j} \qquad (4.3.5b)$$

Where the following shorthand notation is used:

$$\Delta x_i^+ = x_{i+1,j} - x_{i,j} \quad ; \quad \Delta y_i^+ = y_{i+1,j} - y_{i,j}$$

$$\Delta x_i^- = x_{i-1,j} - x_{i,j} \quad ; \quad \Delta y_i^- = y_{i-1,j} - y_{i,j}$$

$$\Delta x_j^+ = x_{i,j+1} - x_{i,j} \quad ; \quad \Delta y_j^+ = y_{i,j+1} - y_{i,j}$$

$$\Delta x_j^- = x_{i,j-1} - x_{i,j} \quad ; \quad \Delta y_j^- = y_{i,j} - y_{i,j-1}$$

Notice that in the above definitions, the subscript 'old' has been dropped from the points at i,j to make the equations easier to read. Also keep in mind that in Figure 4.1 the effect of the quantities a and b on $\vec{V}_1$ and $\vec{V}_2$ have not been taken into consideration.

Adding equations (4.3.5a) and (4.3.5b) gives the Cartesian components of $\vec{R}_{2_{i,j}}$:

$$\vec{R}_{2_{i,j}} = R_x\,\hat{i} + R_y\,\hat{j}$$

where

$$R_x = \frac{1}{2}a\,(\Delta x_i^+ - \Delta x_i^-) + \frac{1}{2}b\,(\Delta x_j^+ - \Delta x_j^-) \qquad (4.3.6a)$$

$$R_y = \frac{1}{2}a\,(\Delta y_i^+ - \Delta y_i^-) + \frac{1}{2}b\,(\Delta y_j^+ - \Delta y_j^-) \qquad (4.3.6b)$$

Although equations (4.3.6a) and (4.3.6b) give the components of point movement in Cartesian coordinates, in this form it is not easy to determine which of these components pulls the point in a particular direction of the physical domain. As a particular example consider a solid boundary located along the $\xi$ = constant direction. Since the $\xi$ direction runs along a constant value of the j

index, it becomes obvious which components of the $\vec{R}_{2_{i,j}}$ vector pull the points away from the surface if the components of the vector are in the $\xi$ and $\eta$ or i and j directions. (The i and j indices do not point in the $\hat{i}$ and $\hat{j}$ directions.) An alternative expression for $\vec{R}_{2_{i,j}}$ as a sum of two vectors in the $\xi$ and $\eta$ directions is therefore highly desirable. Figure 4.2 suggests clearly that the two vectors we are seeking are $\hat{\alpha}$ and $\hat{\gamma}_{i,j}$ which do indeed point along the $\eta$ and $\xi$ directions respectively. A formal definition of the two unit vectors $\hat{\alpha}_{i,j}$ and $\hat{\gamma}_{i,j}$ is:

$$\hat{\alpha}_{i,j} = \frac{1}{\sqrt{\alpha_{i,j}}} (x_\eta \hat{i} + y_\eta \hat{j})_{i,j} \qquad (4.3.7a)$$

$$\hat{\gamma}_{i,j} = \frac{1}{\sqrt{\gamma_{i,j}}} (x_\xi \hat{i} + y_\xi \hat{j})_{i,j} \qquad (4.3.7b)$$

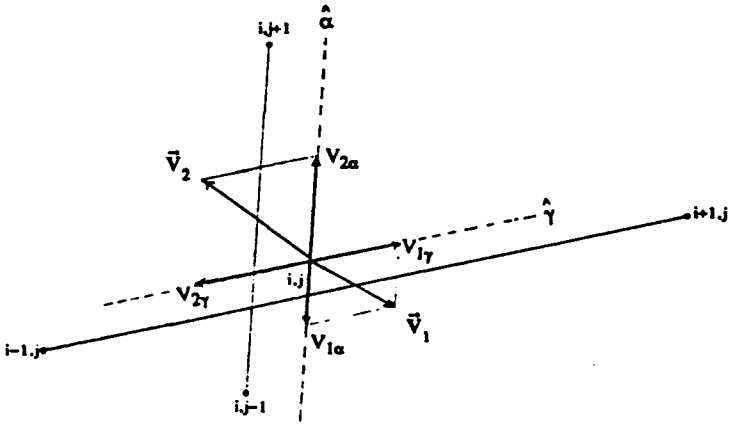



**Figure 4.2**

The quantities $\alpha_{i,j}$ and $\gamma_{i,j}$ are formally defined as:

$$\alpha_{i,j} = (\vec{P}_\eta \cdot \vec{P}_\eta)_{i,j}$$

$$\gamma_{i,j} = (\vec{P}_\xi \cdot \vec{P}_\xi)_{i,j}$$

We can thus write:

$$\vec{R}_{2_{i,j}} = R_x \hat{i} + R_y \hat{j} = (V_{1\gamma} + V_{2\gamma})\hat{\gamma}_{i,j} + (V_{1\alpha} + V_{2\alpha})\hat{\alpha}_{i,j} \qquad (4.3.9)$$

The quantities $V_{1\gamma}$, $V_{2\gamma}$, $V_{1\alpha}$ and $V_{2\alpha}$ are the components of the vectors $\vec{V}_1$ and $\vec{V}_2$ in the and $\hat{\alpha}$ and $\hat{\gamma}$ directions. For arbitrary unit vectors $\hat{\alpha}$ and $\hat{\gamma}$ which make an angle $\theta$ between them, we can write by consulting Figure 4.2:

$$V_{1\gamma} = \vec{V}_1 \cdot \hat{\gamma} - V_{1\alpha} \cos\theta$$

$$V_{1\alpha} = \vec{V}_1 \cdot \hat{\alpha} - V_{1\gamma} \cos\theta$$

similarly:

$$V_{2\gamma} = \vec{V}_2 \cdot \hat{\gamma} - V_{2\alpha} \cos\theta$$

$$V_{2\alpha} = \vec{V}_2 \cdot \hat{\alpha} - V_{2\gamma} \cos\theta$$

The above set of equations can be solved for the four unknowns $V_{1\gamma}$, $V_{2\gamma}$, $V_{1\alpha}$ and $V_{2\alpha}$. However, we mentioned that our grid is orthogonal in the regions in which we are interested. This means the angle $\theta$ is 90°. As a result, the above set of equations reduce to taking the dot product of $\vec{V}_1$ and $\vec{V}_2$ with $\hat{\alpha}_{i,j}$ and $\hat{\gamma}_{i,j}$. Using equations (4.3.5a) and (4.3.5b) along with (4.3.7a) and (4.3.7b), we get:

$$V_{1\alpha} \approx \vec{V}_1 \cdot \hat{\alpha} = \frac{1}{2} a \left(\Delta x_i^+ - \Delta x_i^-\right) \frac{x_\eta}{\sqrt{\alpha_{i,j}}} + \frac{1}{2} a \left(\Delta y_i^+ - \Delta y_i^-\right) \frac{y_\eta}{\sqrt{\alpha_{i,j}}} \qquad (4.3.10)$$

$$V_{1\gamma} \approx \vec{V}_1 \cdot \hat{\gamma} = \frac{1}{2} a \left(\Delta x_i^+ - \Delta x_i^-\right) \frac{x_\xi}{\sqrt{\gamma_{i,j}}} + \frac{1}{2} a \left(\Delta y_i^+ - \Delta y_i^-\right) \frac{y_\xi}{\sqrt{\gamma_{i,j}}} \qquad (4.3.11)$$

$$V_{2\alpha} \approx \vec{V}_2 \cdot \hat{\alpha} = \frac{1}{2} b \left(\Delta x_j^+ - \Delta x_j^-\right) \frac{x_\eta}{\sqrt{\alpha_{i,j}}} + \frac{1}{2} b \left(\Delta x_j^+ - \Delta x_j^-\right) \frac{y_\eta}{\sqrt{\alpha_{i,j}}} \qquad (4.3.12)$$

$$V_{2\gamma} \approx \vec{V}_2 \cdot \hat{\gamma} = \frac{1}{2} b \left(\Delta x_j^+ - \Delta x_j^-\right) \frac{x_\xi}{\sqrt{\gamma_{i,j}}} + \frac{1}{2} b \left(\Delta x_j^+ - \Delta x_j^-\right) \frac{y_\xi}{\sqrt{\gamma_{i,j}}} \qquad (4.3.13)$$

In Figure 4.2, if curvature at the point (i,j) where to have no pulling effect on the grid point at (i,j), the pair of points at (i+1,j) and (i−1,j) should pull the central point along the $\hat{\alpha}_{i,j}$ direction only and similarly the pair of points at (i,j+1) and (i,j-1) should pull it along the $\hat{\gamma}_{i,j}$ direction only.

In other words by setting $V_{1\alpha}$ and $V_{2\gamma}$ equal to zero the pulling effect of curvature will be completely eliminated.

Remember however that we are going to be prudent and try to retain as much of the curvature effect as possible. Also, in certain regions we might want to diminish the effect of curvature selectively. In terms of our equations, the above observation amounts to rewriting equation (4.3.9) in the following form:

$$\vec{R}_{2_{i,j}} = R_x \hat{i} + R_y \hat{j} = (V_{1\gamma} + PV_{2\gamma})\hat{\gamma}_{i,j} + (QV_{1\alpha} + V_{2\alpha})\hat{\alpha}_{i,j} \qquad (4.3.14)$$

In this form, adjusting the values of $P$ and $Q$ allows us to adjust the pulling effect of curvature in each direction independently. But we are not finished! We have so far worked under the assumption that the grid is orthogonal in the regions we are interested in. What happens in other regions? Obviously as the grid deviates from orthogonality, the accuracy of equations (4.3.3a) and (4.3.3b), which formed the basis of our derivation and led to (4.3.14), diminishes. We need a way to compensate for this and provide a smooth transition between the regions which are guaranteed to be orthogonal and those which are not.

*Let the relative motion vector* $\vec{R}_{i,j}$ *be composed of two components, one which satisfies* the full elliptic equations; that is equations (4.1.8a) and (4.1.8b) and is donated $\vec{R}_{1_{i,j}}$ and another which satisfies equation (4.3.10) and is donated $\vec{R}_{2_{i,j}}$, consistent with our previous definitions of these two vectors. Let a scalar $\Phi$ be defined such that it varies smoothly between o and 1. We can write:

$$\vec{R}_{i,j} = \Phi\, \vec{R}_{1_{i,j}} + (1 - \Phi)\, \vec{R}_{2_{i,j}} \qquad (4.3.11)$$

Equation (4.3.11) states that unless $\Phi$ is equal to 0, the solution will not be without the influence of the full elliptic equations, and that if $\Phi$ is equal to 1, the solution depends entirely on $\vec{R}_{1_{i,j}}$.

## 4.4 The GridPro™/pc2000 Computer Program

The above technique of curvature control has been successfully implemented in the GridPro™/pc2000 elliptic grid generation computer program. This program forces the grid to be orthogonal at the boundaries of the domain. In the interior regions of the grid, the orthogonality

condition is not enforced, allowing deviation from orthogonality. In most practical cases the grid stays very close to orthogonal even in the interior of the domain.

Equation (4.3.11) is implemented in GridPro™/pc2000 by letting $\Phi$ vary exponentially between a minimum value at the walls and 1 in the interior regions. The minimum value depends on the topology. For O grids the minimum value is 0.25. The scalars $P$ and $Q$ are also adjusted according to the topology. Again we consider the O grid as an example. Here the direction along the body is the $\xi$ and the direction normal to the body the $\eta$ direction. When an elliptic grid with an O topology is constructed about a body such as an ellipse, the region near the tip shows an intense clustering of points, as seen in Figure 4.3a. This clustering is desirable in the $\xi$ direction, but the grid can be improved by reducing the clustering in the $\eta$ direction. When this is done, the result is the grid in Figure 4.3b. Notice the normal spacing is now much more uniform along the ellipse. The grid in Figure 4.3a shows a significant change in normal spacing between the top part and tip section of the ellipse. The grid in Figure 4.3b was produced using GridPro™/pc2000 by setting $Q$ equal to 0. Figure 3a was produced with all the settings the same except with Q equal to 1. For both cases $P$ was equal to 1.

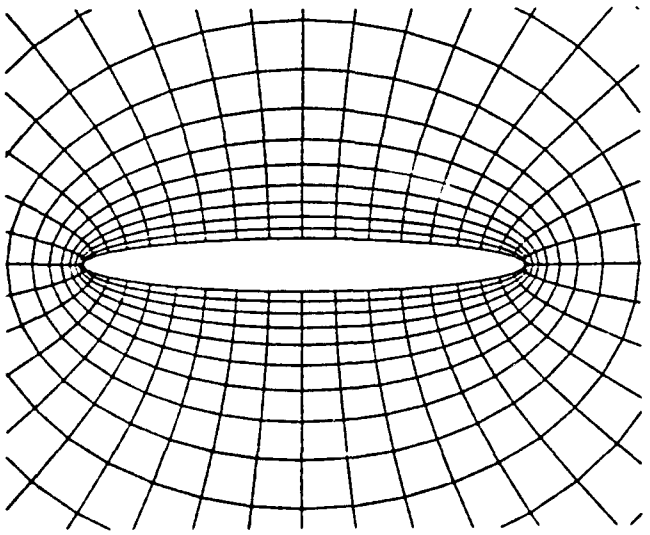


**Figure 4.3a**

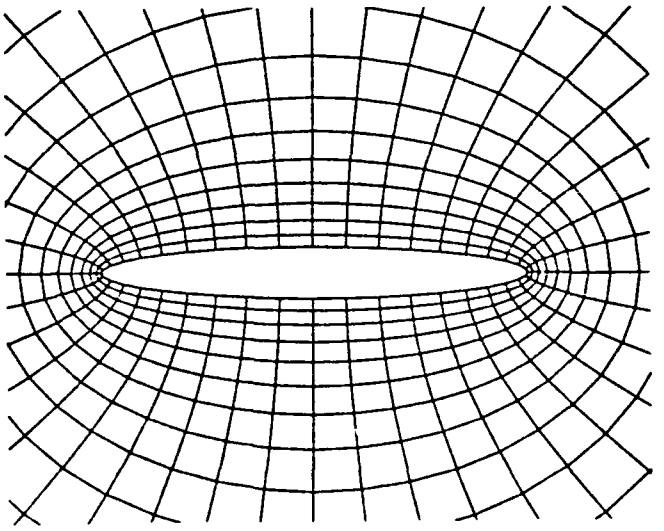


**Figure 4.3b**

## CHAPTER V. SUMMARY

During our Phase II work, (i) we have made the control point form (CPF) of algebraic grid generation more applicable in the real world of grid generation, and (ii) we have developed an effective method to control the grid point distribution of elliptic grids along curved boundaries.

(i) Progress made in the CPF of algebraic grid generation was reported in Chapters II and III. In Chapter II, we reviewed the old blending functions and described how to construct new and more flexible blending functions as well as the results, both for $C^1$- and $C^2$-continuity blending functions. The new blending functions can allow a computer software user to choose the locations of control points arbitrarily. In Chapter III, we discussed how to make the 2D transformation relation given by the CPF to be a bilinear transformation within a quadrilateral (which includes a parallelogram as a special case). This helps us to recapture the general clustering feature of a given grid of arbitrary shape when we use the CPF to regenerate the given grid. These two improvements have been included in one of our grid generation programs called **GridPro/sb**, which can be used to improve the quality of a single-block 3D volumetric grid. In our software **GridPro**/sb3010 and **GridPro**/sb3015, a user can insert or remove a control surface (or a control line in a boundary surface) one at a time. In software **GridPro**/sb3000–**GridPro**/sb3015, a user has the choice of preserving the clustering of an initial grid.

(ii) Progress made in elliptic grid generation was reported in Chapters IV. Our method of controling the grid point distribution near curved boundaries has been implemented in our grid generation code **GridPro**/pc2000. This code produces more uniform grid point spacing along curved boundaries in both the tangential and normal directions.

## REFERENCES

[1] P. R. Eiseman, "Coordinate generation with precise control over mesh properties," J. Comput. Phys. **47**, 331 (1982).

[2] P. R. Eiseman, "High level continuity for coordinate generation with precise controls," J. Comput. Phys. **47**, 352 (1982).

[3] P. R. Eiseman, "A control point form of algebraic grid generation," International Journal for Numerical Methods in Fluids **8**, 1165 (1988).