Iterative circle-inserting algorithm CST3D-OC of truly orthogonal curvilinear grid for coastal or river modelling

H Kim[1], S Lee[1,3], J Lee[1] and H-S Lim[2]

[1]Kookmin University, Seoul, Korea
[2]Korea Institute of Ocean Science and Technology, Ansan, Korea

E-mail: stork@kookmin.ac.kr

Abstract. A geometric method to generate orthogonal curvilinear grid is proposed here. Elliptic partial differential equations have frequently been solved to find orthogonal grid positions, but questions on orthogonality have remained so far. Algebraic methods have also been developed to improve orthogonality, but their applications have been limited to special situations. When two confronting boundary lines of the quadrilateral boundaries are straight, and their positions are known, and we assume that some degree of freedom exists on the other two confronting boundary curves under the condition that the each curve passes through a point, we can assign a set of latitudinal curves in the domain using polynomials. The curves are expected not to fold on their own. The grid positions along longitudinal curves are found by inserting circles between two neighbouring latitudinal curves one by one. If the two curves are straight, the new grid point above the grid point of interest can be found geometrically. This algorithm involves iterations because the curves are not straight lines. The present new algorithm is applied to a domain, and produced almost perfect orthogonality, and similar aspect ratio compared to an existing partial differential equation approach. The present algorithm also can express almost quadrant domain. The present algorithm seems useful for generation of orthogonal curvilinear grids along coasts or rivers. Some example grids are demonstrated.

1. Introduction

Some governing equations for fluid flow use other coordinates from Cartesian coordinates. When boundary shape is not straight, and still flow behaviour near boundary plays an unimportant role, boundary fitted grid becomes more useful. Flows at rivers or coasts are often horizontally two-dimensional, and requires curvilinear grid. Furthermore, if terms for non-orthogonal grid are to be excluded, orthogonality of a curvilinear grid becomes very important, and affects the accuracy of the whole computation.

Methods to generate orthogonal curvilinear grid may be evaluated on three quantities, i.e. orthogonality, aspect ratio, and computational speed of generation. We don't pay attention to the computational speed here. The relative importance of the orthogonality and the aspect ratio depends on the situation or modellers preference, if one value erodes the other value. The uneven aspect ratio has been treated by numerical schemes, while the poor orthogonality cannot be recovered without adopting other governing equations which include terms for non-orthogonality. Here we assume a situation that the orthogonality is the ultimate target, the aspect ratio is the second priority.

Orthogonal grid generation has often been conducted by solving an elliptic partial differential equation (PDE) like Laplace equation or Poisson equation [1-3]. However, there have been arguments that the PDE methods don't guarantee orthogonality of the generated grids. Interim results towards asymptotic solution from the PDE method show very poor orthogonality.

On the other hand algebraic methods may have advantage to generate orthogonal grids due to its inherent mathematical attribute. An algebraic grid generation method by Chibisov et al [4] uses circle-rolling method with invariant radiuses to produce almost orthogonal grid, but their method is limited to situations when orthogonal grid is needed around an obstacle or a boundary, and cannot choose the other outside or inside boundary limit. In that case, their method is not ideal for resolving coastal areas which are clearly defined by surrounding land and offshore boundaries. Conti et al [5] proposed a modified PDE method by including an interpolation, but their results still contain poor orthogonality.

We assume that the distribution of coordinate lines has some degree of freedom as Duraiswami and Prosperetti [6] argued. We divide the lines into longitudinal lines and latitudinal lines for convenience. If we can arrange non-intersecting latitudinal lines at first, then the following longitudinal lines will not intersect at all. We attempt to generate grid points for a quadrilateral domain, two confronting boundaries of which are straight, and each of the other two boundary lines passes through a known position.

2. Circle-inserting grid generation algorithm

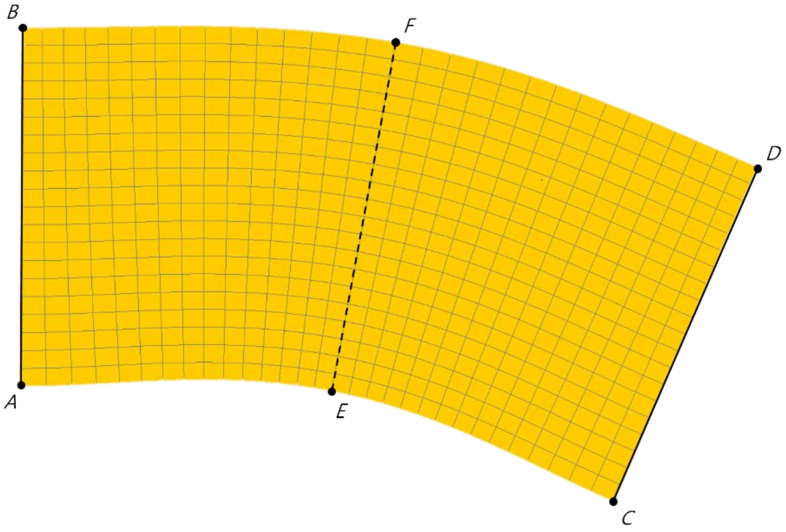
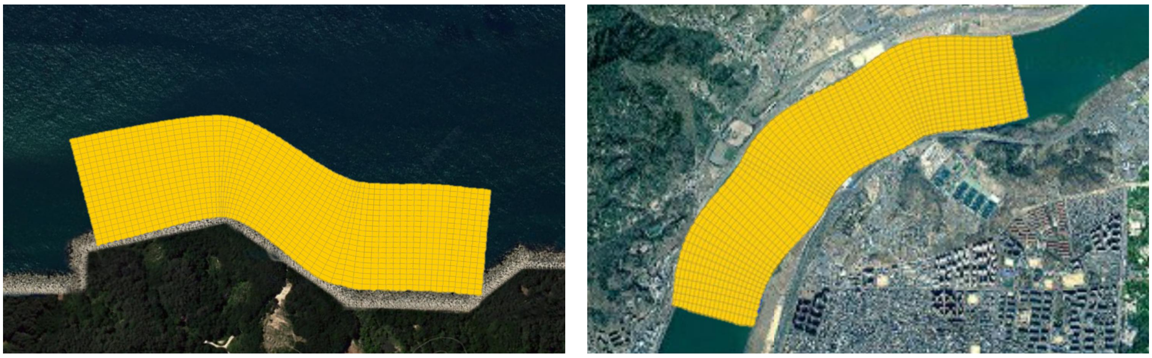


Figure 1. Definition of one domain.



(a) Coast (b) River

Figure 2. Use of orthogonal curvilinear grids.

If a domain is surrounded by two opposite straight lines like figure 1, it is convenient to attach another domain with the same straight line at one end, see figure 2. This is ideal for describing coastal zones or rivers. Block border lines are recommended to sit at relatively straight areas. Separation of a large domain into several small domains is useful if the whole domain is distorted with a wide angle. Even a circular domain like coast surrounding an island can be described by four quadrants.

We start from choosing two straight confronting boundary lines AB and CD, then, we choose another straight line EF between the two boundary straight lines, which provides mid points of the latitudinal lines. We call AB, CD and interim quasi-parallel lines "longitudinal lines", and all other orthogonal lines from AB to CD lines "latitudinal lines". We can choose arbitrary ratio for selection of points along the three straight line segments, AB, CD, and EF. Each latitudinal line has 5 constraints: point positions and gradients at two side boundaries, and a point position between them. Therefore, a fourth order polynomial function for each latitudinal line is adequate to contain these five constraints.

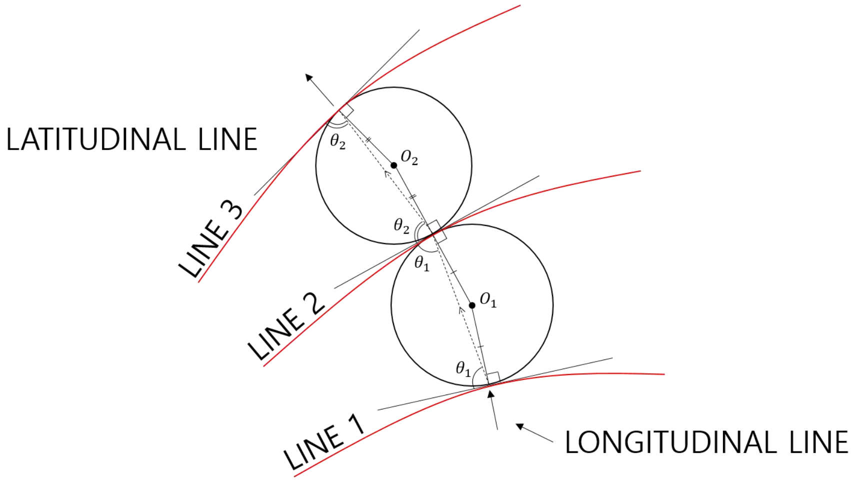


Figure 3. Circle-inserting between curves.

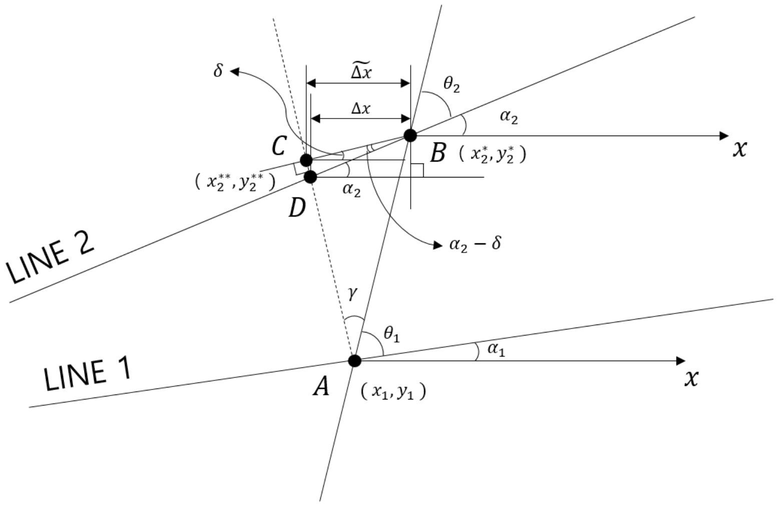


Figure 4. Iterative method of geometric finding of new.

When the latitudinal lines are chosen, longitudinal lines should be found to decide the final grid points. However, we won't try to find any full algebraic equation for a longitudinal line. Instead we choose the grid point positions along the bottom latitudinal line. Now we start to find the first longitudinal line. We insert a circle between the bottom latitudinal line and the next latitudinal line, so that the two latitudinal lines are tangent to the inner circle, and make the two intersection angles equal, e. g. θ_1 or θ_2, see figure 3. If the two latitudinal lines are straight, the new point on the upper latitudinal line can be found geometrically, see figure 4. The interaction point B chosen as the first prediction should be replaced by the correct point D as:

$$\Delta x = \frac{d \sin\gamma \cos\alpha_2}{\cos\delta} \tag{1}$$

$$\gamma = \frac{\theta_1 + \theta_2 - \pi}{2} \tag{2}$$

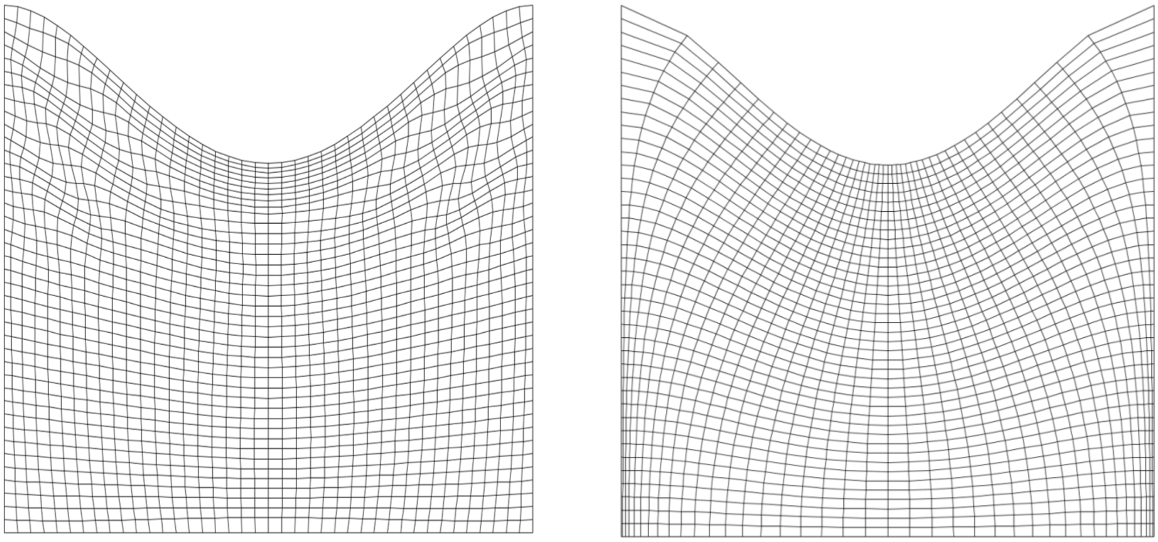
$$\delta = \frac{\alpha_2 - \alpha_1}{2} \tag{3}$$

where α_1 is the gradient of the bottom line, α_2 is the gradient of the upper line, θ_1 is the angle between the bottom latitudinal line and the longitudinal line, and θ_2 is the angle between the upper latitudinal line and the longitudinal line.

Because the lower and upper latitudinal lines are not strictly straight, the above correction is not complete at the first trial, and requires repetition until a given allowance is met. This geometric iteration could also be replaced by other algebraic iteration method like Newton-Rhapson method.

3. Applications

The present algorithm is applied to two simple domains, a square-like one, and a partial quadrant (donut). Domain A is surrounded by three straight lines, and a polynomial curve, see figure 5.



(a) PDE method after 200 iterations; (M=41, N=41) (b) present algorithm; (M=41, N=41)

Figure 5. Domain A (boundary passes (0,0), (1,0), (1,1), (0.5, 0.7), (0, 1)).

An existing PDE method [3] is compared with the present method for Domain A. Four properties are compared for evaluation of the two methods, i.e. the maximum deviation from orthogonality (O_{max}), the mean deviation from orthogonality (O_{mean}), the maximum grid aspect ratio (R_{max}), and the mean grid aspect ratio (R_{mean}).

The maximum deviation from orthogonality is computed by:

$$O_{max} = \max\left(\left|90^{\circ} - \theta_{i,j}\right|\right) \tag{4}$$

The mean deviation from orthogonality is computed by:

$$O_{mean} = \frac{1}{(M-2)(N-2)} \sum_{i=2}^{M-1} \sum_{i=2}^{N-1} \left| 90^{\circ} - \theta_{i,j} \right| \tag{5}$$

The maximum grid aspect ratio is computed by:

$$R_{max} = \max\left(\max\left(\frac{h_{\xi_{i,j}}}{h_{\eta_{i,j}}}, \frac{h_{\eta_{i,j}}}{h_{\xi_{i,j}}}\right)\right) \tag{6}$$

And mean grid aspect ratio is computed by:

$$R_{mean} = \frac{1}{(M-2)(N-2)} \sum_{i=2}^{M-1} \sum_{i=2}^{N-1} \max\left(\frac{h_{\xi_{i,j}}}{h_{\eta_{i,j}}}, \frac{h_{\eta_{i,j}}}{h_{\xi_{i,j}}}\right) \tag{7}$$

$$\theta_{i,j} = \arccos\left(\frac{g_{12}}{\sqrt{g_{11}}\sqrt{g_{22}}}\right)_{i,j} \tag{8}$$

Computed four properties of the two methods are shown in table 1.

Table 1. Comparison of a PDE method and the present method.

	O_{max}	O_{mean}	R_{max}	R_{mean}
PDE	0.474	0.203	2.88	1.43
Present	0.121	0.000252	9.97	1.76

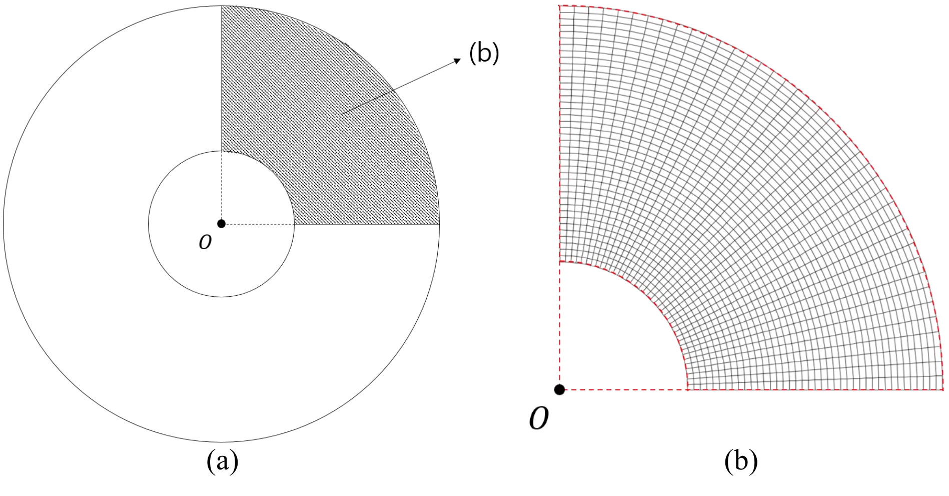
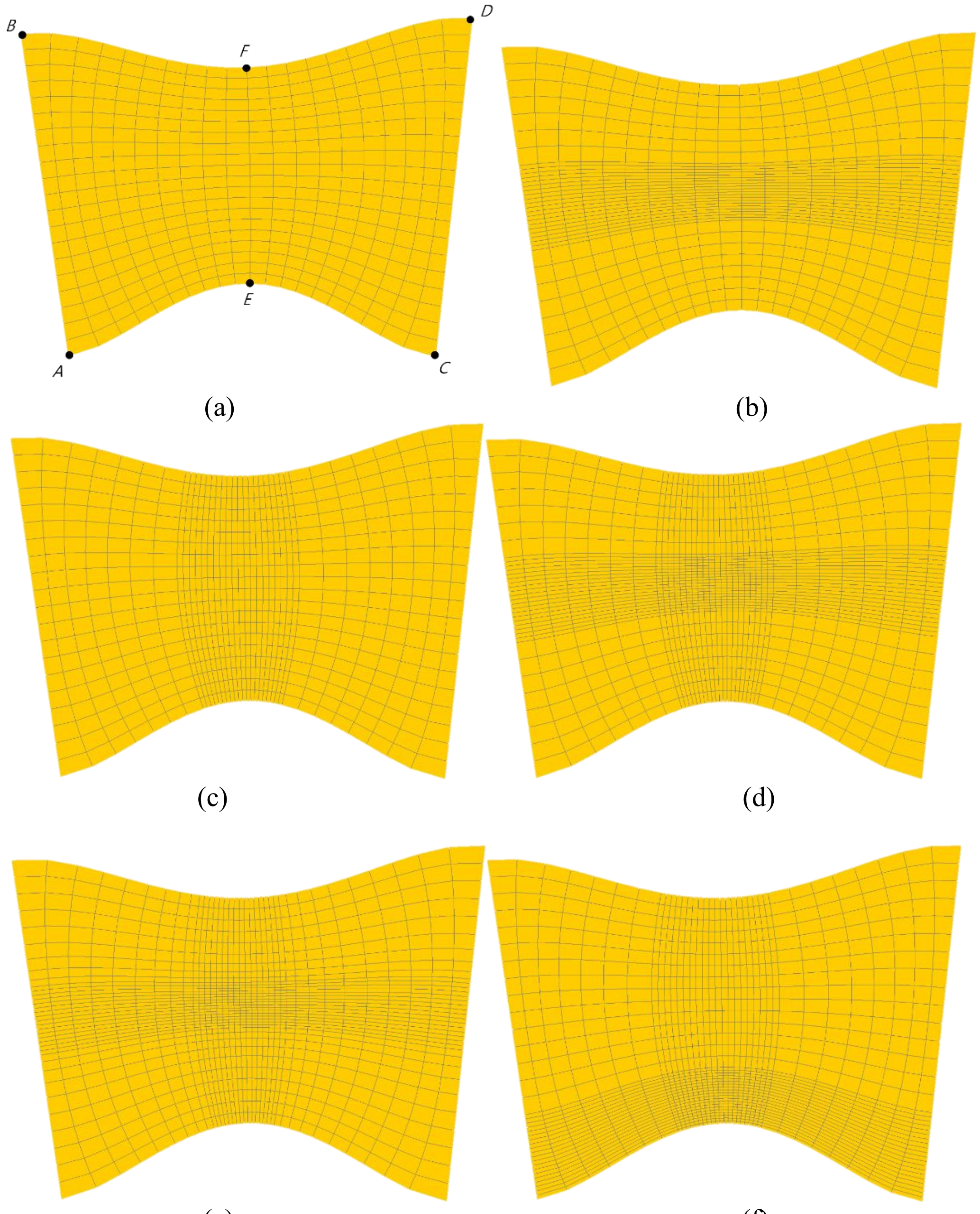


Figure 6. Domain B.

Results demonstrate that the present circle-inserting method produces almost orthogonal grid. Conceptually the present method should produce perfectly orthogonal grid, but small deviation from perfect orthogonality appears due to the finite sizes of the grid cells. On the other hand both methods show similar order of aspect ratios, although the present method produces slightly larger aspect ratios for this specific case. Next the present method was applied to a circular domain, Domain B, see figure 6. The present method reproduces a quadrant, and the maximum error in radius is 0.354%, which is thought sufficiently small. This test result demonstrates a full circular domain can be expressed by four partial quadrants, each of which can be represented by the grid produced by the present method.

The present method is relatively free from choosing the internals latitudinal lines, see figure 7(a). If we want a zoomed zone in the vertical direction, an option of vertical zooming can be used, see figure 7(b). Similarly a horizontal zooming zone can be assigned, see figure 7(c). Zooming on both directions can also be used, see figure 7(d). The unzoomed and zoomed zones are separated, and the size increment experience discontinuity, and this problem can be overcome by using the transitional zooming, see figure 7(e). If a zoomed zone is attached to a border line like a coastline, the zoomed zone can situated at a side, see figure 7(f). If a domain has complicated geometry, it can be manually split and joined, see figure 7(g). The discretization between two adjacent block domains can be smoothly transitioned, see figure 7(h), and a blog site [7].



(a)

(b)

(c)

(d)

(e)

(f)

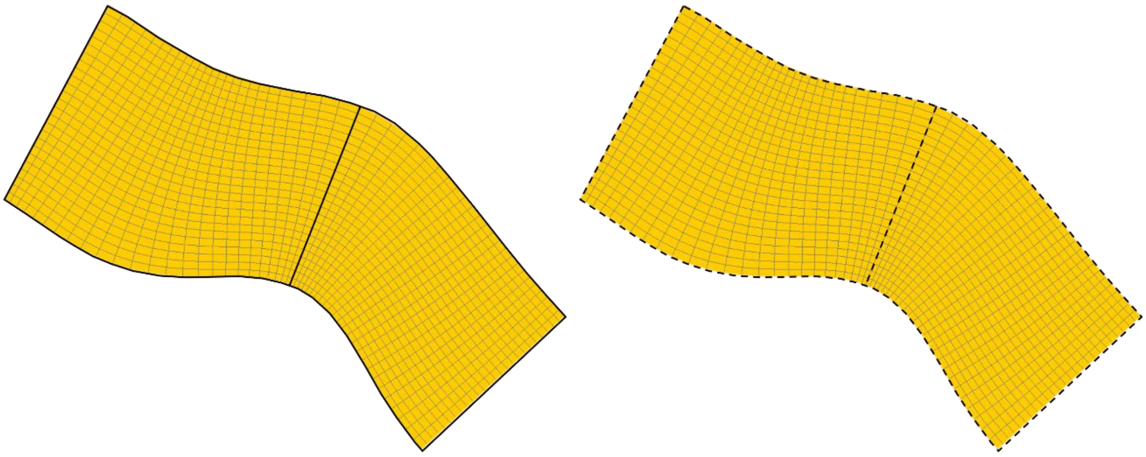


Figure 7. Various applications.

4. Conclusions

A geometric iterative method to generate orthogonal curvilinear grid was proposed. The present algorithm guarantees orthogonality of the grid due to circle-inserting concept, i.e. the grid generated is almost orthogonal as inner circles are tangential to neighbouring latitudinal lines. The grid points are found successively from one end to the other end of the latitudinal lines. The method can also provide reasonable aspect ratios compared to an existing PDE method. The method is useful for resolving coastal or river zones which could be expressed as combination of Domain A or B.

Acknowledgments

This is a result of the research project titled “Development of Coastal Erosion Control Technology” funded by the Ministry of Oceans and Fisheries, Korea.

References

[1] Ryskin G and Leal L G 1983 Orthogonal mapping *J. Comput. Phys.* **50** 71-100

[2] Thompson J F, Soni B K and Weatherill N P 1999 *Handbook of Grid Generation* (Boca Raton: CRC Press)

[3] Akcelik V, Jaramaz B and Chattas O 2001 Nearly orthogonal two-dimensional grid generation with aspect ratio control *J. Comput. Phys.* **171** 805-21

[4] Chibisov D, Ganzha V, Mayr E W and Vorozhtsov E V 2005 Generation of orthogonal grids on curvilinear trimmed regions in constant time computer algebra in scientific computing 2005 *Lecture Notes in Computer Science* **3718** 105-14

[5] Conti C, Morandi R and Spitaleri R M 2005 An algebraic-elliptic algorithm for boundary orthogonal grid generation *Appl. Math. Comput.* **162** 15-27

[6] Duraiswami R and Prosperetti A 1992 Orthogonal mapping in two dimensions *J. Comput. Phys.* **98** 254-68

[7] Him K 2017 *Coastal Sediment Transport in 3 Dimensions* Retrieved from http://blog.naver.com/ cst3d