# CHAPTER 3

# POLYGONAL FINITE ELEMENT METHODS

## 3.1 Brief background in polygonal FEM

The terminology of polygonal FEM used here is more subjected to irregular or arbitrary polygonal elements applied for sub-division of the domain for FEM calculation. The common utilized elements, such as triangular and rectangular elements, are categorized as polygonal elements, i.e. 3-gons and 4-gons, respectively. In linear triangular FEM, the interpolation function is based on the barycentric coordinates as explained in Chapter 2 while for linear rectangular FEM the interpolation function is based on Lagrangian shape function as mentioned in [15], which will not be discussed more in this paper.

An effort to generalize barycentric coordinates from triangular elements to arbitrary polygonal ($n$-gonal) elements has been taken so far and yields some proposed formulations as explained in [16]. One of the formulations, with similar properties as the triangular barycentric coordinate, is the Wachspress shape function proposed by Eugene Wachspress in 1975. This function is simplified by Mark Meyer, et.al. in 2002 as found in [16] and is used as generalized barycentric coordinates of convex irregular $n$-gons. In the following times, some formulations for general polygonal interpolation are proposed especially for polygonal interpolation functions in FEM, such as Laplace shape function, mean value coordinates, metric coordinates, and MAXENT shape function [13,18,21]. J. Warren constructed an extension of Wachspress's formulations to convex polytopes in arbitrary dimensions in 1996 as cited from [16, 19].

The utilization of polygons with $n>4$ are not much employed yet in finite element computations using this Wachspress shape function especially in solving electromagnetic problems. While applications of this construction of barycentric coordinates on general polygons are already for parameterization, surface/geometric modeling, and CAD [16].

## 3.2 Polygonal shape functions

When the elements used in FEM become polygons ($n$-gons) in generalized irregular forms with $n = 3$ and $n > 3$ as in Figure 3.1, the shape function is changed and the method now is called a polygonal FEM [13, 16-21]. A polygon is a closed plane figure made up of several line segments that are joined together where its sides do not cross each other and exactly two sides meet at every vertex. Many works try to build a shape function for an irregular polygon and yield some proposed shape functions based on area coordinate, angle, or distance calculations as mentioned in the previous sub-chapter. Generally, all of them can be applied on convex elements while some of them can be assigned for concave elements, such as metric coordinates and mean value coordinates. Recently, Timo Euler starts the polygonal FEM on electromagnetic problems

without concerning the accuracy using mean value coordinates for non-concave elements [28].

In this wok, the performance of the Wachspress shape function applied for convex elements in polygonal FEM is observed which is based on area coordinates as well as in linear triangular FEM. The more simple and straightforward calculation for its shape function lead the choice of this Wachspress shape function as the interpolation function in FEM here. Therefore, this polygonal FEM can be compared to linear triangular FEM so that the performance of the generalized interpolation function of arbitrary convex elements (polygons) can be investigated.

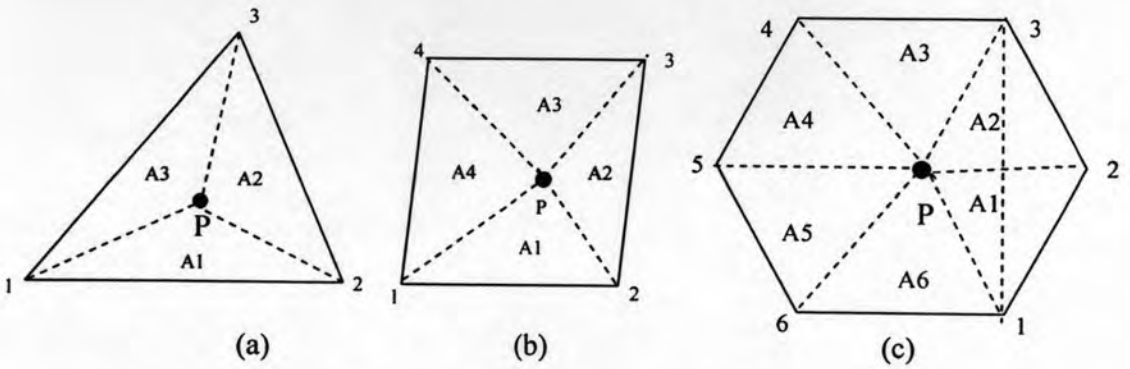



Figure 3.1 (a) A triangular ($n$=3), (b) a quadrilateral ($n$=4) and (c) a hexagonal ($n$=6) elements.

### 3.2.1 Constructing the Wachspress shape functions

The shape function on any convex domain can be constructed using Wachspress' formulation. This new barycentric formulation with a barycentre (centre of element mass) $p$ is simple, local, and easy to implement [16]. If $n = 3$ then the formula reduces to the classical equations of area coordinates. If $p$ is strictly within the polygon element $Q$, then the non-normalized barycentric coordinate called as a weight function for node $j$, i.e. $\omega_j$, is given as follows [16, 18]:

$$\omega_j = \frac{A(\phi_{j-1},\phi_j,\phi_{j+1})}{A(\phi_{j-1},\phi_j,p)A(\phi_j,\phi_{j+1},p)} \tag{3.1}$$

where $\bar{\phi} = \sum_{i=1}^{n} N_i^e \phi_i^e$ and $N_i^e = \dfrac{\omega_i^e}{\sum_{j=1}^{n} \omega_j^e}$. (3.2)

For example, the polygonal elements are rectangular elements Figure 3.1as in Figure 3.1b with $n = 4$. In this case, the approximate unknown will have 4 terms in its expansion, i.e.

$$\bar{\phi} = \sum_{i=1}^{n} N_i^e \phi_i^e = \sum_{i=1}^{4} N_i^e \phi_i^e = \left[N^e\right]^T \left\{\phi^e\right\} \tag{3.3}$$

where

$$N_1^e = \frac{\omega_1^e}{\sum_{j=1}^{4}\omega_j^e},\quad N_2^e = \frac{\omega_2^e}{\sum_{j=1}^{4}\omega_j^e},\quad N_3^e = \frac{\omega_3^e}{\sum_{j=1}^{4}\omega_j^e} \text{ and } \quad N_4^e = \frac{\omega_4^e}{\sum_{j=1}^{4}\omega_j^e}. \tag{3.4}$$

with

$$\omega_1^e = \frac{A(\phi_4,\phi_1,\phi_2)}{A(p,\phi_4,\phi_1)A(p,\phi_1,\phi_2)} \tag{3.5}$$
$$= \frac{2[(x_1y_2 - x_2y_1) + (y_1 - y_2)x_4 + (x_2 - x_1)y_4]}{[(x_4y_1 - x_1y_4) + (y_4 - y_1)x + (x_1 - x_4)y][(x_1y_2 - x_2y_1) + (y_1 - y_2)x + (x_2 - x_1)y]}$$

$$\omega_2^e = \frac{A(\phi_1,\phi_2,\phi_3)}{A(p,\phi_1,\phi_2)A(p,\phi_2,\phi_3)} \tag{3.6}$$
$$= \frac{2[(x_2y_3 - x_3y_2) + (y_2 - y_3)x_1 + (x_1 - x_2)y_1]}{[(x_1y_2 - x_2y_1) + (y_1 - y_2)x + (x_2 - x_1)y][(x_2y_3 - x_3y_2) + (y_2 - y_3)x + (x_3 - x_2)y]}$$

$$\omega_3^e = \frac{A(\phi_2,\phi_3,\phi_4)}{A(p,\phi_2,\phi_3)A(p,\phi_3,\phi_4)} \tag{3.7}$$
$$= \frac{2[(x_3y_2 - x_4y_3) + (y_3 - y_4)x_2 + (x_4 - x_3)y_2]}{[(x_2y_3 - x_3y_2) + (y_2 - y_3)x + (x_3 - x_2)y][(x_3y_4 - x_4y_3) + (y_3 - y_4)x + (x_4 - x_3)y]}$$

$$\omega_4^e = \frac{A(\phi_3,\phi_4,\phi_1)}{A(p,\phi_3,\phi_4)A(p,\phi_4,\phi_1)} \tag{3.8}$$
$$= \frac{2[(x_4y_1 - x_1y_4) + (y_4 - y_1)x_3 + (x_1 - x_4)y_3]}{[(x_3y_4 - x_4y_3) + (y_3 - y_4)x + (x_4 - x_3)y][(x_4y_1 - x_1y_4) + (y_4 - y_1)x + (x_1 - x_4)y]}$$

Finally, the built shape function consists of high order polynomials, which results high order interpolation function in a rational form inside the element but remains linear on the edges of the elements.

For $n = 4$, the order of the result polynomial interpolation function based on the Wachspress shape function formulation become complete quadratic as in (3.9) for each numerator and denominator,

$$N_i^e(x,y)\big|_{n=4} = \frac{a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2}{b_1 + b_2x + b_3y + b_4x^2 + b_5xy + b_6y^2} \tag{3.9}$$

where the same order polynomial interpolation before can be achieved by a 6-node triangular element with mid side nodes using Lagrange's interpolation functions [24].

Another example is an element with $n$=5 (a pentagon) in which both numerator and denominator of the shape function become complete $C^o$-cubic polynomials as given in (3.10).

$$N_i^e(x,y)\Big|_{n=5} = \frac{a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2 + a_7x^3 + a_8x^2y + a_9xy^2 + a_{10}y^3}{b_1 + b_2x + b_3y + b_4x^2 + b_5xy + b_6y^2 + b_7x^3 + b_8x^2y + b_9xy^2 + b_{10}y^3} \tag{3.10}$$

From these two examples of shape functions, increasing the number of $n$ will increase the order of the interpolation function so that the higher accuracy can be obtained. The expected order of the polynomial will be equal to $n$-2 proved by the derivation of the shape functions for $n = 4$ and $n = 5$ above.

The unique of this generalized polygonal shape function is that the formulation become linear as used in a linear ($n$=3) triangular element. It can be proved as follows. For $n = 3$, the example of the element is shown in Figure 3.1a with

$$N_i^e(x,y)\Big|_{n=3} = \frac{\omega_i}{\omega_1 + \omega_2 + \omega_3}; \text{ for } i = 1,2,3 \tag{3.11}$$

where

$$\omega_1 = \frac{A(1,2,3)}{A(p,3,1)A(p,1,2)}; \ \omega_2 = \frac{A(1,2,3)}{A(p,1,2)A(p,2,3)} \text{ and } \omega_3 = \frac{A(1,2,3)}{A(p,2,3)A(p,3,1)} \tag{3.12}$$

Substituting those weight functions into the shape function formulation gives

$$N_i^e(x,y) = \frac{\omega_i}{A(1,2,3)\times\left(\frac{1}{A(p,3,1)A(p,1,2)} + \frac{1}{A(p,1,2)A(p,2,3)} + \frac{1}{A(p,2,3)A(p,3,1)}\right)} \tag{3.13}$$

As an example, for $i$ =1,

$$N_1^e(x,y) = \frac{\frac{A(1,2,3)}{A(p,3,1)A(p,1,2)}}{A(1,2,3)\times\left(\frac{1}{A(p,3,1)A(p,1,2)} + \frac{1}{A(p,1,2)A(p,2,3)} + \frac{1}{A(p,2,3)A(p,3,1)}\right)} \tag{3.14}$$

$$= \frac{\frac{1}{A(p,3,1)A(p,1,2)}}{\left(\frac{A(p,2,3) + A(p,3,1) + A(p,1,2)}{A(p,3,1)A(p,1,2)A(p,2,3)}\right)} \tag{3.15}$$

$$= \frac{A(p,2,3)}{A(p,2,3) + A(p,3,1) + A(p,1,2)} \tag{3.16}$$

$$= \frac{A(p,2,3)}{A(1,2,3)} \tag{3.17}$$

The last equality becomes the same as written in (2.47) for such $i$ which is used as a linear interpolation function for linear triangular elements. In summary, this can be represented by a Pascal triangle shown in Figure 3.2 [24].

| Pascal triangle : | Degree, number of terms and related number of $n$ : | | |
|---|---|---|---|
| 1 | 0 (constant) | 1 term | - |
| x y | 1 (linear) | 3 terms | $n$=3 |
| $x^2$ xy $y^2$ | 2 (quadratic) | 6 terms | $n$=4 |
| $x^3$ $x^2y$ $xy^2$ $y^3$ | 3 (cubic) | 10 terms | $n$=5 |
| $x^4$ $x^3y$ $x^2y^2$ $xy^3$ $y^4$ | 4 (quartic) | 15 terms | $n$=6 |
| $x^5$ $x^4y$ $x^3y^2$ $x^2y^3$ $xy^4$ $y^5$ | 5 (quintic) | 21 terms | $n$=7 |

Figure 3.2 A Pascal triangle in relationship with the number of complete polynomial degree and the number of term in x-y plane used in the rational polynomial interpolants of Wachspress shape functions.

This shows the flexibility of the Wachspress shape function when it is applied to any $n$-gonal elements for $n$ equal to or greater than 3 which it can be linear or higher order depending on the number of $n$. This characteristic of the shape function promises the conformity among elements with a different number of $n$ since the shape function maintain the linearity on the boundary of the elements while the variation of orders of the interpolation function occurs inside the elements. Due to the larger area that can be covered by such polygonal element, less number of elements and nodes is needed in the discretization with the promising higher order of the elements which is expected to give higher accuracy. The placement of $n$-gonal elements on the surface mesh must be considered and designed carefully based on the behavior of the analyzed system. An example of plotted Wachspress shape functions is shown in Figure 3.3.

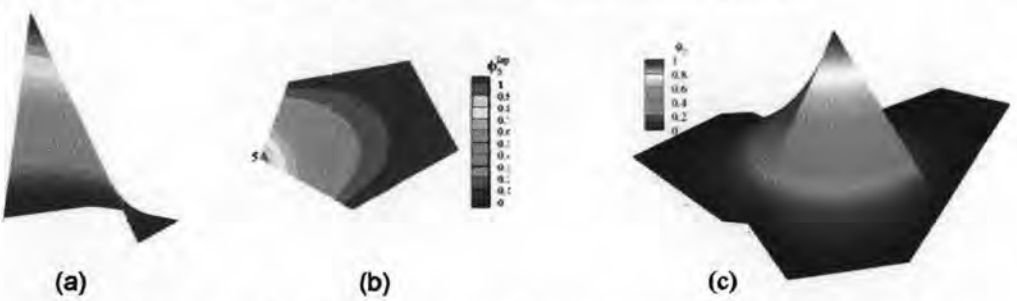



Figure 3.3 An example of Wachspress shape function on a pentagonal ($n$=5) element from (a) side view, (b) top view and (c) conforming to neighboring elements [13, 18].

This shape function conforms over arbitrary convex polygonal elements although the order of elements may be different among each other. The condition can be accomplished since the characteristic of the shape function follows the property as mentioned in Chapter 2.3. Therefore, this shape function maintains the linearity on the element boundary and increase the element order inside as shown in Figure 3.3c.

Before polygonal FEM is available, single type of elements is utilized in the domain discretization. In order to approximate the solution more accurately, a refinement mesh in a certain part of the domain is taken. Another way is to increase the order of the element so that a higher order of interpolation function is obtained. In case of the domain discretization, polygonal FEM give a simpler, higher order interpolation function and

more flexible in choosing types of elements compared to the conventional FEM. The FEM formulation is taken still using the Galerkin's method as in the linear triangular FEM except for the integration. In the previous chapter, the linear triangular FEM employs the analytical integration for the FEM formulation that constructs the element matrices. In polygonal FEM, the analytical integration might bring a complicated formulation due to the polynomial of the denominator in the shape function so that the integration is taken using numerical integration (quadrature), which is explained as follows.

### 3.2.2 Numerical integration of the weak form in polygonal elements

After the shape function is constructed, the integration of the weak formulation in (2.63) over the domain is taken to find the approximate solutions. A function $f$ is defined to present the weak formulation used in the numerical integration where

$$\iint f(x,y)dxdy = \iint\left(\frac{\partial\left[N_i^e\right]}{\partial x}\frac{\partial\left[N_i^e\right]^T}{\partial x}+\frac{\partial\left[N_i^e\right]}{\partial y}\frac{\partial\left[N_i^e\right]^T}{\partial y}\right)dxdy-\left(\frac{\omega}{c}\right)^2\iint\left[N_i^e\right]\left[N_i^e\right]^T\varepsilon_r dxdy \tag{3.18}$$

Numerical integration on $V_k$ which is $k^{\text{th}}$ polygonal element is approached by partitioning it into triangles and then integrate them using standard quadrature rules on triangles [18] where the element is the original physical element as illustrated in Figure 3.4.

$$\iint_{V_k} f(x,y)dxdy = \sum_{e=1}^{n}\iint_{V_k,\Delta_e} f(x,y)dxdy = \sum_{e=1}^{n}\int_0^1\int_0^{1-\xi} f(\xi,\eta)\left|J_1^e\right|d\xi d\eta \tag{3.19}$$

where $|J_1^e|$ is determinant of Jacobian matrix on element $e$ as follows

$$\left|J_1^e\right| = \begin{vmatrix} \partial\bar{x}/\partial\xi & \partial\bar{y}/\partial\xi \\ \partial\bar{x}/\partial\eta & \partial\bar{y}/\partial\eta \end{vmatrix} = \left(\partial\bar{x}/\partial\xi\,\partial\bar{y}/\partial\eta - \partial\bar{x}/\partial\eta\,\partial\bar{y}/\partial\xi\right) \tag{3.20}$$

There is an isoparametric mapping operation which converts the plane of the weak formulation integration from $x$-$y$ plane to $\xi$-$\eta$ plane using this relation by using the same interpolation function that are used to approximate the unknown [14] and the function of Jacobian matrix is shown in (3.21) which is already proved in [32].

$$dxdy = \left|J_1^e\right|d\xi d\eta \tag{3.21}$$

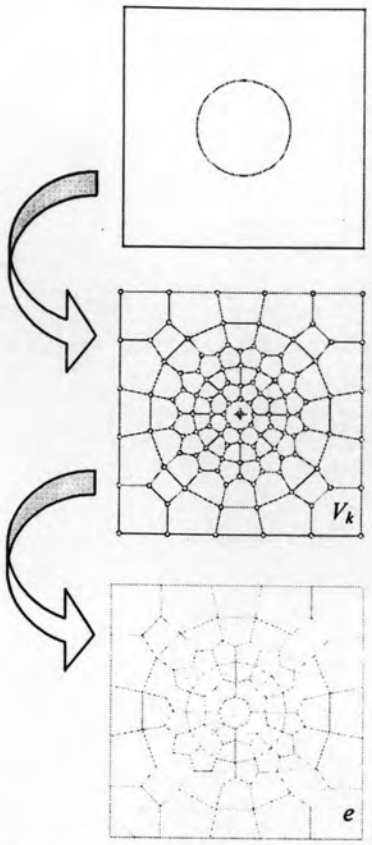



Figure 3.4 An illustration of discretization of (a) a unit cell using (b) arbitrary polygonal elements ($n$>3) including (c) the sub-triangulations.

So, for a given quadrature point, $\bar{x} = \sum_{i=1}^{3} N_i^e(\xi,\eta)\bar{x}_i$ is used where the equation for $N_i^e(\xi,\eta)$ is the same as the finite element shape function for a three node triangle as in Figure 3.5.

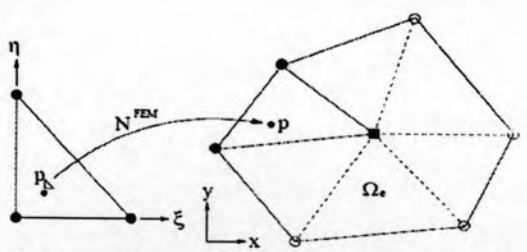



Figure 3.5 A numerical integration scheme using a partition of the physical element [18].

For general number of gauss points $q$'s, the formulation for quadrature becomes:

$$I = \sum_{k=1}^{K} \sum_{e=1}^{E_k} \iint_{\Delta_e} f_k(x,y)dxdy \tag{3.22}$$

$$= \sum_{k=1}^{K} \sum_{e=1}^{E_k} \int_0^1 \int_0^{1-\xi} f_k^e(\xi,\eta)\left|J^e(\xi,\eta)\right| d\eta d\xi \tag{3.23}$$

$$= \sum_{k=1}^{K} \sum_{e=1}^{E_k} \int_0^1 \int_0^{1-\xi} g_k^e(\xi,\eta) d\eta d\xi \tag{3.24}$$

$$= \sum_{k=1}^{K} \sum_{e=1}^{E_k} \sum_{q=1}^{Nq} w_q g_k^e(\xi_q,\eta_q) \tag{3.25}$$

where $g_k^e(\xi,\eta) = f_k^e(\xi,\eta)\left|J^e(\xi,\eta)\right|$ and $w_q$ is the appropriate weight for gauss point $q$. Symbol $I$ is an integration for each polygonal element that yields the approximate value for the unknown. After the integration is done, the eigensystem can be established so that the eigenvalues can be evaluated. The used data of gauss points for quadrature are shown in Table 3.1.

Table 3.1 Gauss quadrature's points [34]

| Number of gauss points | $\xi_q$ | $\eta_q$ | $\frac{1}{2}$weight, $w_q$ |
|---|---|---|---|
| 1 | 0.3333333333 | 0.3333333333 | 0.5000000000 |
| 3 | 0.6666666667 | 0.1666666667 | 0.1666666667 |
| | 0.1666666667 | 0.6666666667 | 0.1666666667 |
| | 0.1666666667 | 0.1666666667 | 0.1666666667 |
| 4 | 0.6000000000 | 0.2000000000 | 0.2604166667 |
| | 0.2000000000 | 0.6000000000 | 0.2604166667 |
| | 0.2000000000 | 0.2000000000 | 0.2604166667 |
| | 0.3333333333 | 0.3333333333 | -0.2812500000 |
| 6 | 0.4459484909 | 0.4459484909 | 0.1116907948 |
| | 0.4459484909 | 0.1081030182 | 0.1116907948 |
| | 0.1081030182 | 0.4459484909 | 0.1116907948 |
| | 0.8168475730 | 0.0915762135 | 0.0549758718 |
| | 0.0915762135 | 0.8168475730 | 0.0549758718 |
| | 0.0915762135 | 0.0915762135 | 0.0549758718 |
| 7 | 0.7974269854 | 0.1012865073 | 0.0629695903 |
| | 0.1012865073 | 0.7974269854 | 0.0629695903 |
| | 0.1012865073 | 0.1012865073 | 0.0629695903 |
| | 0.4701420641 | 0.4701420641 | 0.0661970764 |
| | 0.4701420641 | 0.0597158718 | 0.0661970764 |
| | 0.0597158718 | 0.4701420641 | 0.0661970764 |
| | 0.3333333333 | 0.3333333333 | 0.1125000000 |

The position of the gauss points are shown in Figure 3.6.

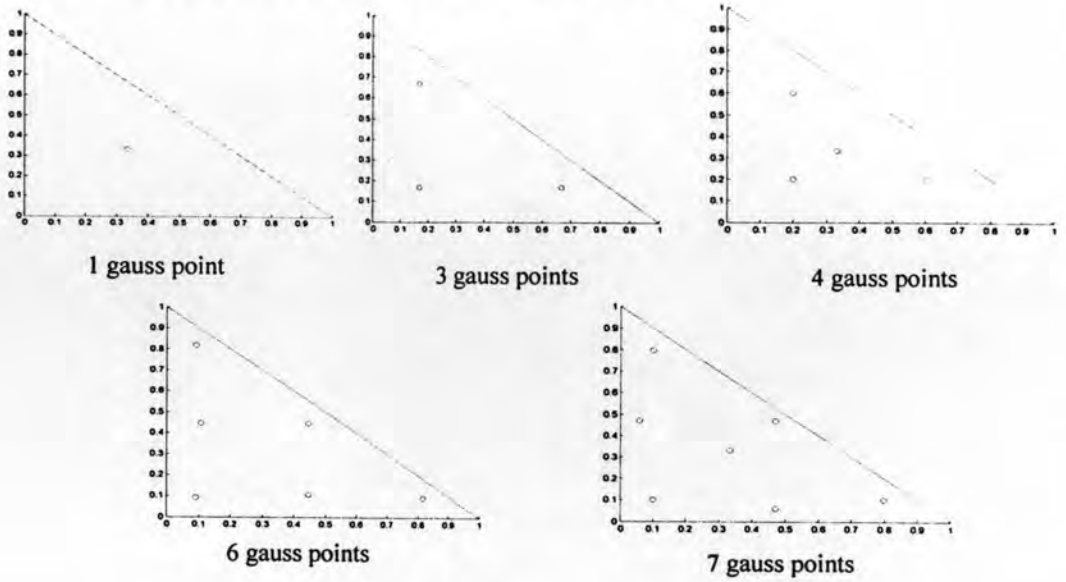



Figure 3.6 Positions of used gauss points based on Table 3.1

### 3.2.3 Construction of a polygonal mesh

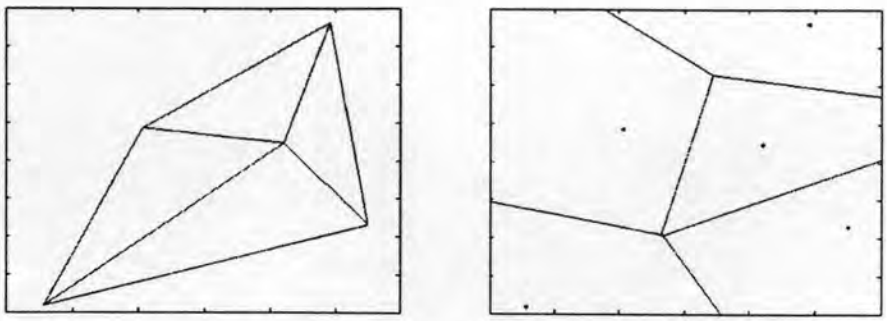
In modeling the system to be analyzed, the physical nature of the problem must be understood earlier in order to make a simplification without loosing the importance information or characteristic as in [24]. In this case, choosing the unit cell as the calculation domain is an efficient and sufficient way in analyzing the band gap characteristic of 2D PCs due to the periodicity of the system.

A polygonal mesh is a group of polygons, which are connected together by shared vertices. No edge should pass through a polygon. The mesh should not contain any errors such as doubled vertices, edges, or faces. It is also important that the mesh becomes a manifold, which means that it does not contain holes or singularities (locations where two distinct sections of the mesh are connected by a single vertex).

To construct a mesh which consists of elements with $n$=3 or $n$=4 is simplified by the available mesh generator software such as pdetooi in MATLAB, GiD, and Plaxis. In order to build a mesh of general polygonal elements with combination of any $n$-gonal elements, some difficulties will be faced because there is still no mesh generator, which can generate such a mesh simultaneously. A function in MATLAB named *voronoi*() can generate such general polygonal mesh known as a Voronoi diagram as the duality of Delaunay triangulation as mentioned in [13] shown in Figure 3.7.

The principle in constructing the Voronoi diagram is similar to Weigner Seitz cell algorithm described in Chapter 2. Unfortunately, although it looks well-defined mesh, some errors can exist in the created polygonal mesh when the numbers of nodes in the triangulation is greater such as collision nodes and extremely short length of edges. Moreover, the coordinates of nodes on the outer boundary of the domain are defined as infinity so that they need to be defined first manually. Another problem is that this

function is still not adaptive to be applied on a domain, which consists of different types of materials such as the unit cell in 2D PCs. More efforts are still needed in order to make the polygonal mesh data ready to use including data of the element matrix.



(a) (b)

Figure 3.7 An example of a duality between (a) Delaunay triangulation consists of five nodes and (b) Voronoi diagram results 5irregular polygonal elements.

The primitive way is generating the mesh manually using mesh generator such as GiD and Plaxis to create the polygonal elements where in Plaxis, each polygonal element is called a *cluster*. Another method to create polygonal elements here is combining some generated triangle elements to be an arbitrary convex polygonal element by doing a mesh conversion. Common tools for mesh generator normally build a polygonal mesh from triangular or quadrilateral elements. In fact, such in computer graphic field of study, several methods, and algorithms are built to generate arbitrary polygonal meshes such as in [25]. Woefully, due to the main objectives of this work and limitation time of study, investigation of generating arbitrary polygonal mesh related to this topic research may become one of the future works.

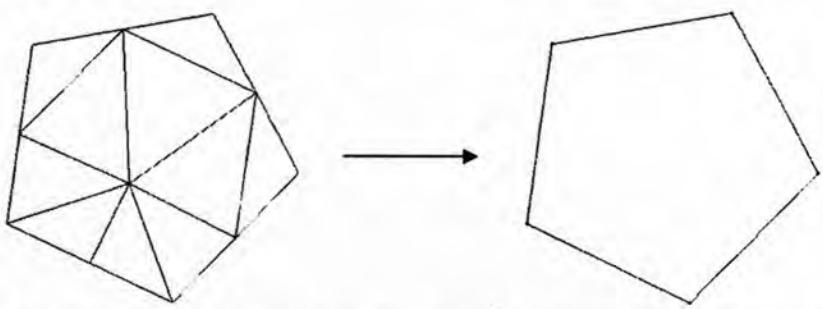


Figure 3.8 A mesh conversion from triangular elements to a polygonal element (pentagon, $n$=5)

Since a triangulation is needed to be drawn in each cluster due to the numerical integration, those mesh generators can make it at once. In another way, the triangulation can be made simply by taking an area from a pair of adjacent vertices and a created node inside the cluster.

In order to observe the performance of the method, the polygonal mesh can be build straightforwardly from rectangular elements where the available mesh generator can make it simultaneously.

### 3.2.4 Derivation of polygonal shape functions for programming

From (2.65) and (2.66), the equalization can be made as follows,

$$\left[A^e\right]=\left[\iint\left(\frac{\partial\left[N_i^e\right]}{\partial x}\frac{\partial\left[N_i^e\right]^T}{\partial x}+\frac{\partial\left[N_i^e\right]}{\partial y}\frac{\partial\left[N_i^e\right]^T}{\partial y}\right)dxdy\right];\ \text{for } i=1,2,..,n \tag{3.26}$$

and

$$\left[B^e\right]=\left\{\iint\left[N_i^e\right]\left[N_i^e\right]^T dxdy\right\};\ \text{for } i=1,2,...,n \tag{3.27}$$

Here, the shape function $N_i^e$ is the Wachspress shape function formulated by (3.1) and (3.2). In order to make it suitable for programming language, the simplification must be taken. Since the formulation in (3.26) consists of the derivation of the shape function, the quotient rule and the chain rule formulation should be applied.

If $v(x)=\dfrac{g(x)}{h(x)}$, then the quotient rule formulation becomes

$$\frac{\partial v(x)}{\partial x}=\frac{h(x).\dfrac{\partial g(x)}{\partial x}-g(x).\dfrac{\partial h(x)}{\partial x}}{h^2(x)} \tag{3.28}$$

while the chain rule is formulated as follows

$$\begin{aligned}\frac{\partial v(x)}{\partial x}&=\frac{\partial v(x)}{\partial\left(g(x)\right)}\frac{\partial g(x)}{\partial x}+\frac{\partial v(x)}{\partial\left(h(x)\right)}\frac{\partial h(x)}{\partial x}\\&=\frac{\partial\left(\dfrac{g(x)}{h(x)}\right)}{\partial\left(g(x)\right)}\frac{\partial g(x)}{\partial x}+\frac{\partial\left(\dfrac{g(x)}{h(x)}\right)}{\partial\left(h(x)\right)}\frac{\partial h(x)}{\partial x}\end{aligned} \tag{3.29}$$

Considering the term of $\dfrac{\partial N_i^e(x,y)}{\partial x}$ for each $i$ and assuming that $y$ is constant at this moment, the combination of (3.28) and (3.29) is taken for

$$\begin{aligned}v(x)&=N_i^e(x,y)\Big|_{y=cons\tan t}\\&=N_i^e(x).\end{aligned} \tag{3.30}$$

Applying (3.28) to (3.30),

$$\begin{aligned}\frac{\partial N_i^e(x)}{\partial x}&=\frac{\partial\left(\dfrac{f_1(x)}{f_2(x)}\right)}{\partial x}\\&=\frac{f_2(x).\dfrac{\partial f_1(x)}{\partial x}-f_1(x).\dfrac{\partial f_2(x)}{\partial x}}{f_2^{\,2}(x)}\end{aligned} \tag{3.31}$$

where

(3.32)

$f_1(x) = \omega_i(x)$ as in (3.1) and

$$f_2(x) = \sum_{i=1}^{n} \omega_i^{\,e} \text{ as in (3.2).} \tag{3.33}$$

By taking a generalization of the formulation for (3.32), for each $i$

$$A_1(x) = A(\phi_{i-1}, \phi_i, p) \tag{3.31}$$

$$A_2(x) = A(\phi_i, \phi_{i+1}, p) \tag{3.32}$$

$$A_{edge}(x) = A(\phi_{i-1}, \phi_i, \phi_{i+1}). \tag{3.33}$$

Since $A_{edge}(x)$ is a constant, $A_{edge} = A_{edge}(x)$.

So, taking the derivation of $f_1(x)$ in (3.31) yields

$$\frac{\partial f_1(x)}{\partial x} = \frac{\partial \omega_i}{\partial x} \tag{3.34}$$

$$\frac{\partial\left(\frac{A_{edge}}{A_1(x)A_2(x)}\right)}{\partial x} = A_{edge}\frac{\partial\left(\frac{1}{A_1(x)A_2(x)}\right)}{\partial x}$$

$$= -A_{edge}\frac{A_2(x)\frac{\partial A_1(x)}{\partial x} + A_1(x)\frac{\partial A_2(x)}{\partial x}}{\left(A_1(x)A_2(x)\right)^2} \tag{3.35}$$

and

$$\frac{\partial f_2(x)}{\partial x} = \frac{\partial\left(\sum_{i=1}^{n} \omega_i^{\,e}\right)}{\partial x} = \frac{\sum_{i=1}^{n} \partial\left(\omega_i^{\,e}\right)}{\partial x} \tag{3.36}$$

$$= \sum_{i=1}^{n} \frac{\partial\left(\omega_i^{\,e}\right)}{\partial x} \tag{3.37}$$

The derivation over $y$ is taken by the same manner and will give the same formulation but respect to $y$.