IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 15, Issue 6 Ser. IV (Nov – Dec 2019), PP 65-75
www.iosrjournals.org

Side based Automatic Mesh generation scheme for a general convex domain with quadrilaterals

Rina Paul[*, a, †], M. S. Karim[a, ‡],Md. Sadekur Rahman[, §]
[a]*Department of Mathematics, Shahjalal University of Science and Technology, Sylhet 3114, Bangladesh*

Abstract: *This article includes an automatic mesh generation scheme foran arbitrary convex domain constituted by straight lines or curves employing lower or higher-order quadrilateral finite elements.First, we develop the general algorithm for h- and p- version meshes, which require the information of sides of the domain and the choice of the order as well as the type of elements.The method also allows one to form the desired fine mesh by providing the number of refinements. Secondly, we develop the MATLAB program based on the algorithm that provides all the valuable and needful outputs of the nodal coordinates, relation between local and global nodes of the elements, and displays the desired meshes. Finally, we substantiate the suitability and efficiency of the scheme through the demonstration of several test cases of mesh generation. We firmly believe that the automatic h- and p- version mesh generation scheme employing the quadrilateral elements will find immense application in the FEM solution procedure.*

Keywords: *Finite element method, Mesh generation, Quadrilateral elements,Convex domain, h- version mesh refinement, p-version mesh refinement;*

Date of Submission: 16-**1**1-2019 Date of Acceptance: 02-12-2019

I. Introduction

Finite element method, the technique of structural analysis, was initially developed as an extension of the standard structural analysis procedure and was intended for application to the design of advanced aeroplane structures. Gradually the versatility of the method and its underlying rich mathematical basis for application in nonstructural areas was recognized by others. As its range of application was being extended rapidly in the early 1960s, it became apparent that the finite element method is essentially a particular discretization procedure which can be employed in the solution of a wide range of field problems. Thus it has much greater importance than just as a tool for structural analysis [1]. Nowadays, in many industrial, medical, and economic applications, the modelling and numerical simulation of the complex system play a crucial role. Mathematically, such a system can be described by partial differential equations (PDEs). For example, heat flow in materials or human body parts, torsion beam suspension in a car, aerodynamic properties of an aeroplane, or determination of option prices in finance are the problems that can be described as PDE and further can be solved through finite element analysis. In contrast, most of these PDEs are likely unsolvable analytically.

In FEM, the continuous domain of the problem is discretized by using a series of simple geometric forms, which are called finite elements. The governing relations on the entire continuous domain are valid on each of these elements. Under this supposition, the approximate solution for the entire continuous domain of the problem can be obtained by using trial functions, which are also called the shape function. FEM then transforms the governing differential equation of the problem domain into an algebraic system of equations, which can then be solved easily by known numerical methods.

Discretization of whole problem domain into simpler parts has many advantages such as- Accurate representation of complex geometry, Inclusion of different material properties, Easy representation of the total solution, Capture of local effects et cetera. [2]. So, the discretization of the geometry of the problem domain, which is called meshing, is one of the essential steps in the procedure of solving PDEs that are described from various real-life problems.

Two types of mesh refinements can be seen in general, h- and p- version mesh refinement. In contrast, h-version mesh refinement means that the size of elements is reduced in order to increase the accuracy of the solution. On the other hand, p-version mesh refinement increases the polynomial order of the element shape functions. It was illustrated based on a linear elastic fracture mechanics problem that sequences of finite element solutions based on the p-version converge faster than sequences based on the h-version bySzabóandMehta [3]. In the case of p-version meshes, based on having internal node/nodes, two types of elements are seen there, Lagrange type and serendipity type elements. The necessity of meshing elements with any of the given types depends upon the purpose needed to serve to solve a real-life problem. Various studies [4, 5] show that the accuracy obtained using the Lagrange type element is higher than that of serendipity type elements. However,

due to longer time length, Lagrange type elements are not preferred to use for commercial purposes. Till today various mesh generation schemes have been developed [6-11]. Most of the techniques dealt with generating structured and unstructured h- version meshes. According to a study conducted on the ins and outs of mesh generation, properties of a proper mesh are – (a) Filling the space (b) Non-overlapping (c) Conforming mesh (d) High-quality elements. Further, since no mathematically sound procedure for obtaining the 'best' mesh distribution is available till now [12], more research is needed to obtain the best procedure.

The necessity, as discussed above, motivates us to develop a new, more uncomplicated technique of generating all quadrilateral meshes. In our proposed automatic mesh generation scheme, we have provided an algorithm and developed a computer code following the algorithm to generate both h-version and p-version quadrilateral element meshes of any arbitrary convex domains according to user's choice. For the proposed scheme, the code uses a side-based mesh generation technique where only the coordinates of the vertices are needed to provide. Then other internal nodes of all the elements, element connectivity, are calculated, formed in the usual way. Finally, it provides all the nodal coordinates and displays the mesh of the domain. If the convex domain contains one or more curved sides, the functions that describe those sides should be provided along with the coordinates of vertices of that portion. It is anticipated that the code and its utilization are more straightforward, uncomplicated, and hence, it may satisfy the requirement of h- and p- version mesh generation with quadrilateral elements.

II. Outline of the proposed mesh generation procedure

In this section, we wish to illustrate the basic idea briefly for generating h- and p- version meshes with quadrilateral elements.

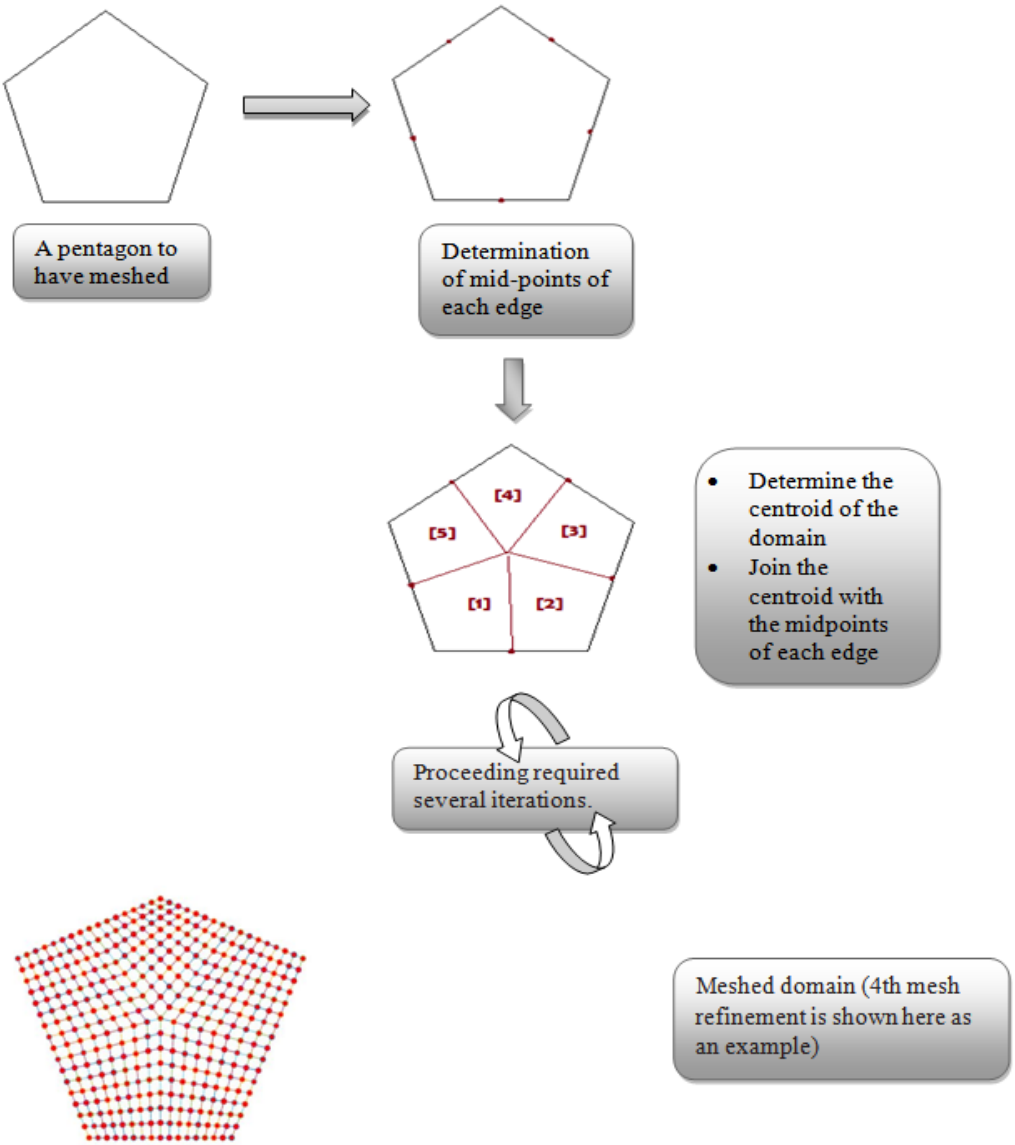


Figure – 1: A simple diagram showing steps of meshing a domain

Let us consider any arbitrary convex domain $\mathbf{C_N}$enclosed with **N**number of sides, which may be straight or curve. Let **M** be the number of curved sides, which can be expressed as functions$\mathbf{F_1}, \mathbf{F_2}, \ldots \ldots \ldots \mathbf{F_M}$. Evaluation of the midpoints$\boldsymbol{M_{e_1}}, \boldsymbol{M_{e_2}}, \boldsymbol{M_{e_3}}, \ldots \boldsymbol{M_{e_{N-M}}}$ of $\mathbf{N-M}$edges(straight sides) and points $\boldsymbol{M_{F_1}}, \boldsymbol{M_{F_2}}, \ldots \ldots \boldsymbol{M_{F_M}}$ of **M**curved sides is done first, and then the centroid of the domain$\mathbf{C_N}$is calculated. Joining all the midpoints/points on the curve with the centroid, we get the domain mesh with **N** number of quadrilateral elements. Continuing the process according to as the necessity, we will get a fine mesh refinement of the domain with more quadrilateral elements. Each refinement will give four times the number of quadrilateral elements than the previous mesh, as shown in Fig-1. Meshing with higher-order elements, i.e., p- version meshes, requires similar procedure and calculation of internal nodes of each element. Accordingly, the element connectivity, relation between the local and global nodes are needed to form, and coordinates are calculated in the usual way.

2.1 Algorithm for h-version & p-version mesh refinement of an arbitrary convex domain

We present here an algorithm for meshing a convex polygonal domain C_N with N sides (i.e., N vertices) to develop a computer code in MATLAB.

Algorithm.

Input:
NE←Required number of elements,
OE← Required order of elements,
QUAD← A1× 4 zero matrix (If the domain itself is a quadrilateral), otherwise QUAD is assigned with 1×4 non-zero matrix.
Nodes← A$N \times 2$ non-zero matrix whose entries are the x- and y-coordinatesof vertices.
Type← 1 (For Lagrange), 2(For Serendipity)
MeshConvex (QUAD, Nodes, NE, OE, Type)

Step 1: Assign **RQ**, **CQ**, **RN**, **CN**
respectively row and column of
QUAD and **Nodes** array.

Step 2: Repeat step 3 to 5 for **I**=1 to **RQ**

Step 3: Calculate the centroid of **QUAD**
Furthermore, compare it with **Nodes** array.

Step 4: Calculate **I**th**QUAD** mid nodes
Furthermore, compare with **Nodes** array.

Step 5: Calculation of nodal coordinates without repetition, i.e., if no repetition is found, increase the index of array**Nodes** by one.

Step 6: Based on calculated centroid and **I**th**QUAD** mid nodes, **I**th**QUAD** element is split into fourquadrilateral elements.

Step 7: If the row of **QUAD** is less than **NE** go to step 1

Step 8: If **OE**=2 and **Type** = Lagrange, call subroutine **Mesh2L (QUAD, Nodes)**, else call subroutine
Mesh2S (QUAD, Nodes),
If **OE**=3 and **Type** = Lagrange, call subroutine **Mesh3L (QUAD, Nodes)** else call subroutine
Mesh3S (QUAD, Nodes)

Step 9: Print **QUAD** and **Nodes** array
Step 10: Draw the figure of the discretized domain
Mesh2L(QUAD, Nodes)

Step 1: Assign **RQ**, **CQ**, **RN**, **CN**
respectively row and column of **QUAD** and **Nodes** array.

Step 2: Repeat step 3 to 5 for **I**=1 to **RQ**

Step 3: **Temp** ←Mid-node of **QUAD(I).**

Step 4: If any entry of **Temp** belongs to **Nodes** array,
Return to Step 2.
Else **Nodes** will be updated with new entry **Temp**and the index of **Nodes** will be increased by one.

Step 5: Calculate the centroid of **QUAD(I)**

Step 6: Based on calculated internal nodes from step 4 and 5, every 4-noded linear **QUAD(I)** becomes 9-noded quadratic element.

Mesh3L(QUAD, Nodes)

Step 1: Assign **RQ**, **CQ**, **RN**, **CN** respectively row and column of **QUAD** and **Nodes** array.

Step 2: Repeat step 3 to 5 for **I**=1 to **RQ**

Step 3: **Temp** ←Nodes that divide edges of **QUAD(I)** into 1:2 and 2:1 proportion respectively**.**

Step 4: If any entry of **Temp** belongs to **Nodes** array,
Return to Step 2.
Else **Nodes** will be updated with new entry **Temp**and the index of **Nodes** will be increased by one.

Step 5: Calculate the Nodes that divide the parallel lines (created by each corresponding internal nodes of **QUAD(I)**calculated in step 4) into 1:2 and 2:1 proportion, respectively.

Step 6: Based on calculated internal nodes from step 4 and 5, every 4-noded linear **QUAD(I)** becomes 16-noded cubic element.

Based on the above algorithm, a complete computer code in MATLAB is developed for meshing the arbitrary convex domain. Two subroutines Mesh2S and Mesh3S, which mesh domain with serendipity type elements, are excluded due to its immense similarity with the other two subroutines Mesh2L and Mesh3L, respectively. Instead, in Mesh2L and Mesh3L, central internal nodes of an element are calculated, which are not necessary in case of the elements of serendipity type.

2.2 Required input, output, and illustration

To illustrate the application of the algorithm, we consider some domains of convex types for experimental interest. So, some of the domains are convex with polygonal boundary, and others are convex domains having one or more curved sides. The nodal coordinates of vertices describe the straight sides, whereas the vertices and the curve functions can describe the curved sides. So, the required inputs for executing the MATLAB program for meshing domains of types discussed above are as follows –

(a) For convex domains with polygonal boundary, required inputs are – Nodal coordinates of the domain's vertices, the order of the elements under consideration for meshing the domain, number of mesh refinements (user's choice), type of elements, i.e., Lagrange or serendipity.

(b) For convex domains with curve side boundary - Nodal coordinates of the domain's vertices, Number of curve sides, expression of curve functions, boundary nodes in the curvature part, the order of the elements under consideration, number of mesh refinements (user's choice), type of elements, i.e., Lagrange or serendipity.

The output of the program includes coordinates of nodes of the meshed domain, representation of elements using the connectivity of nodes, and a visual illustration of the meshed domain. We have demonstrated some cases of h- version (Figure – 2–6) and p- version (Figure – 7) meshing of convex domains acquired by executing the developed MATLAB code based on the technique discussed in the previous section. Convex domains of all types have been considered here. If the domain types are categorized, some categories can be seen as follows –
(1) Convex domain with polygonal boundary (Figure – 2)
(2) Convex domain having both straight and curved boundaries. (Figure – 3, Figure – 4)
(3) Convex domain having only curved boundaries, i.e., the domain enclosed by curve/curves. (Figure – 5, Figure – 6)

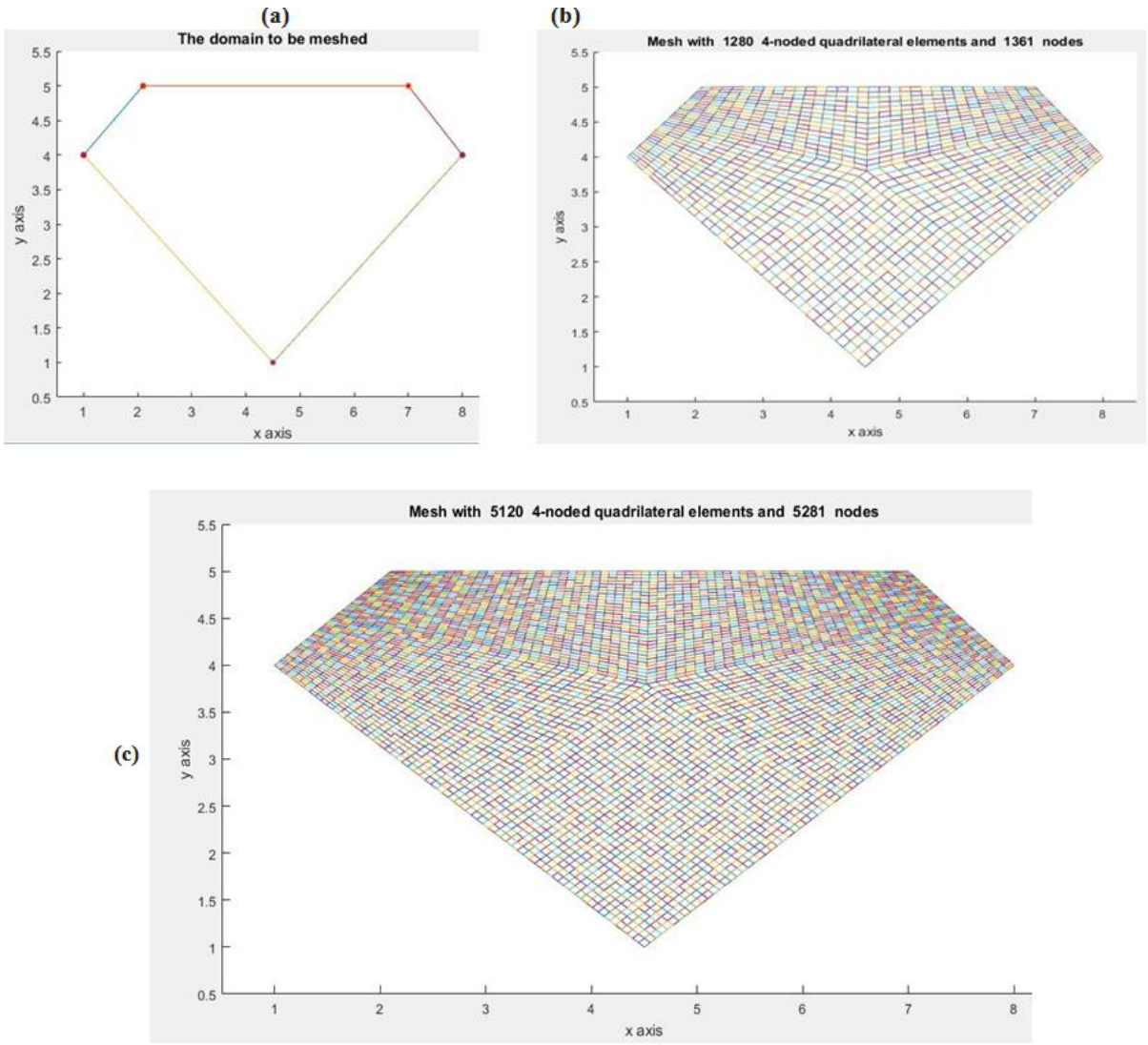


Figure-2: Mesh of a convex polygon (a) Initial domain (b) 1st(c) 5th(d) 6th h-version mesh refinement

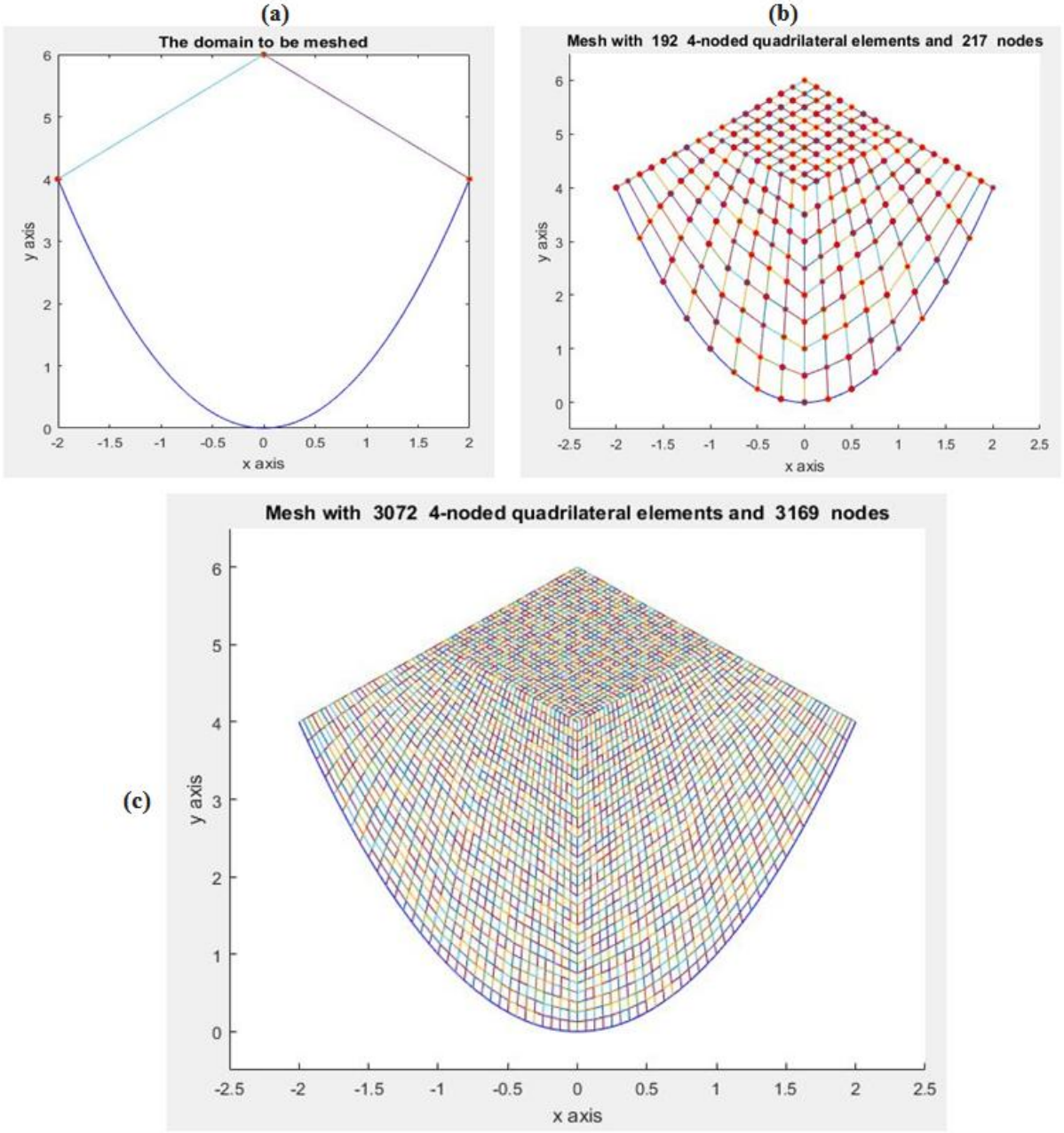


Figure-3: Mesh of a convex domain with 2 straight and one curve sides(a) Initial domain (b) 4th(c) 6thh- version mesh refinement

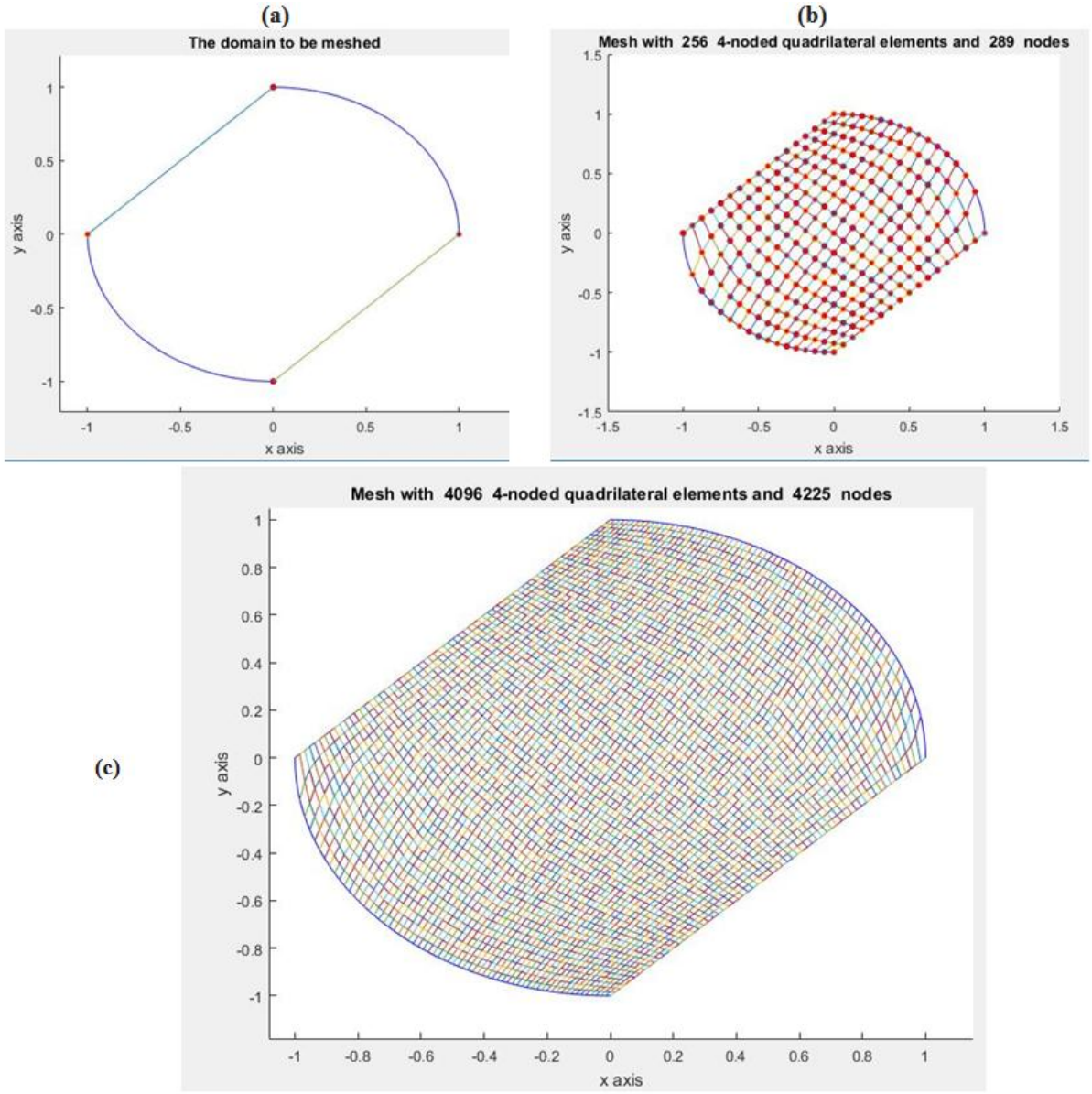
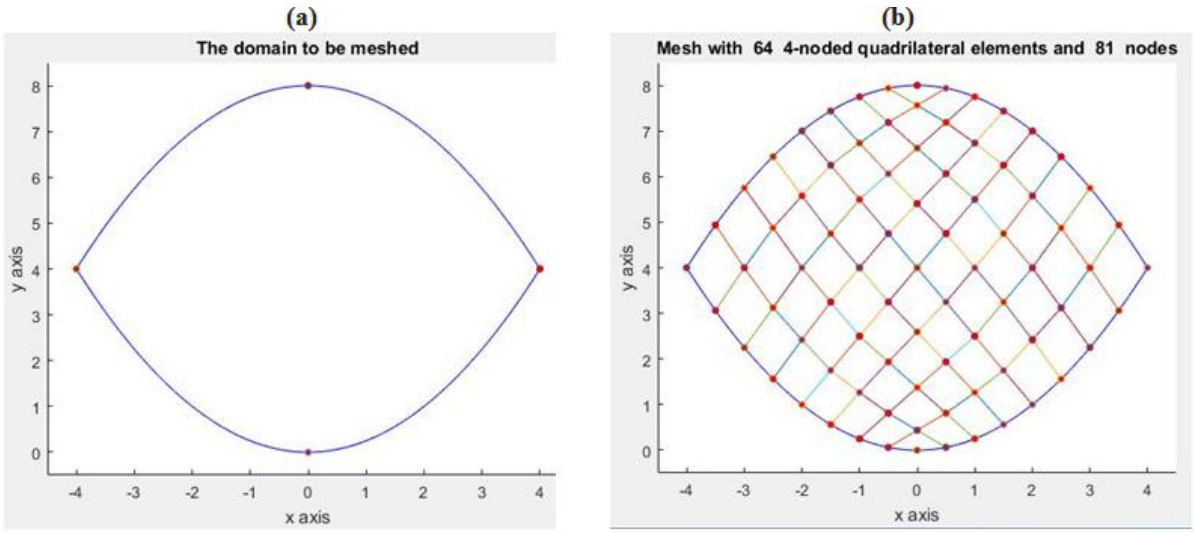


Figure-4: Meshof a convex domain with 2 straight sides and 2 curve sides of different function (a) Initial domain (b) 4th(c) 6thh- version mesh refinement



(c)

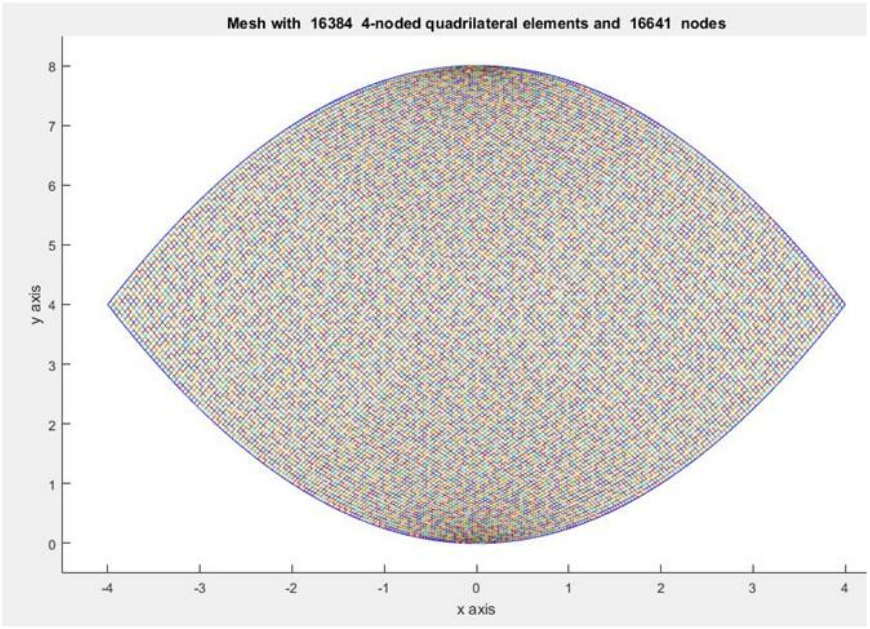
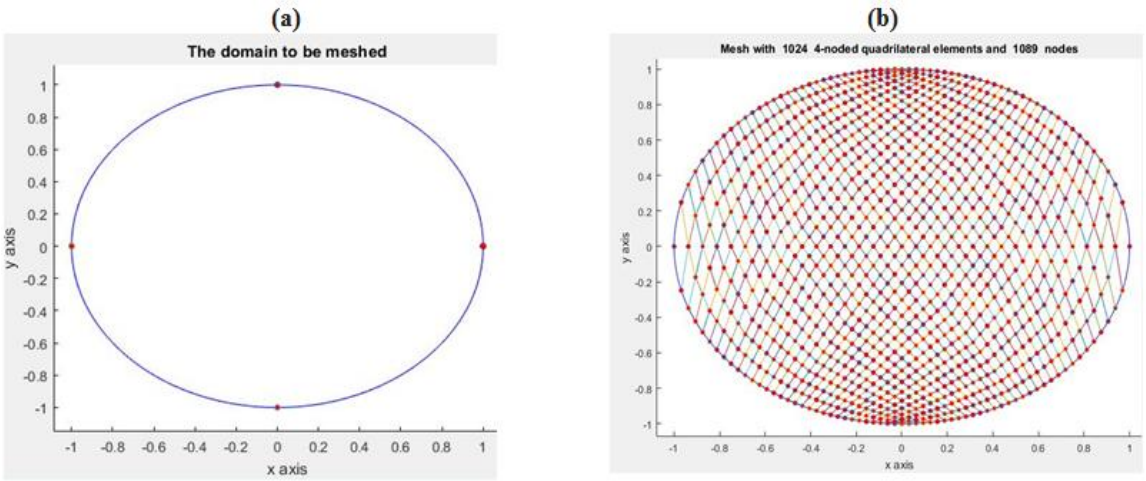


Figure-5: Mesh of a convex domain enclosed by curves(a) Initial domain (b)3rd (c) 7th h- version mesh refinement



(c)

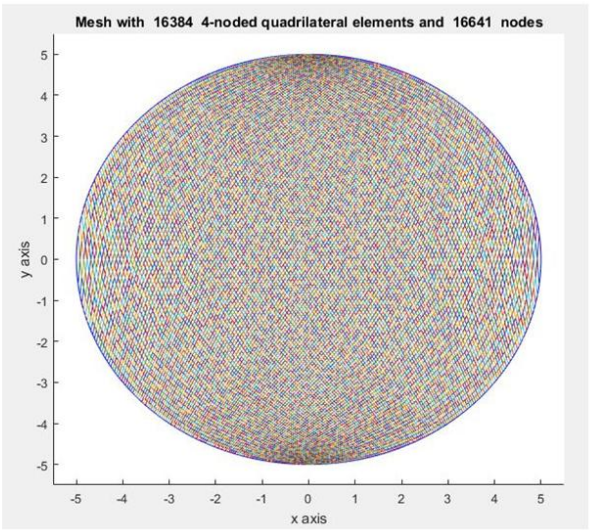


Figure-6: Mesh of a convex domain enclosed by curves(a) Initial domain (b)4th (c) 7thh- version mesh refinement

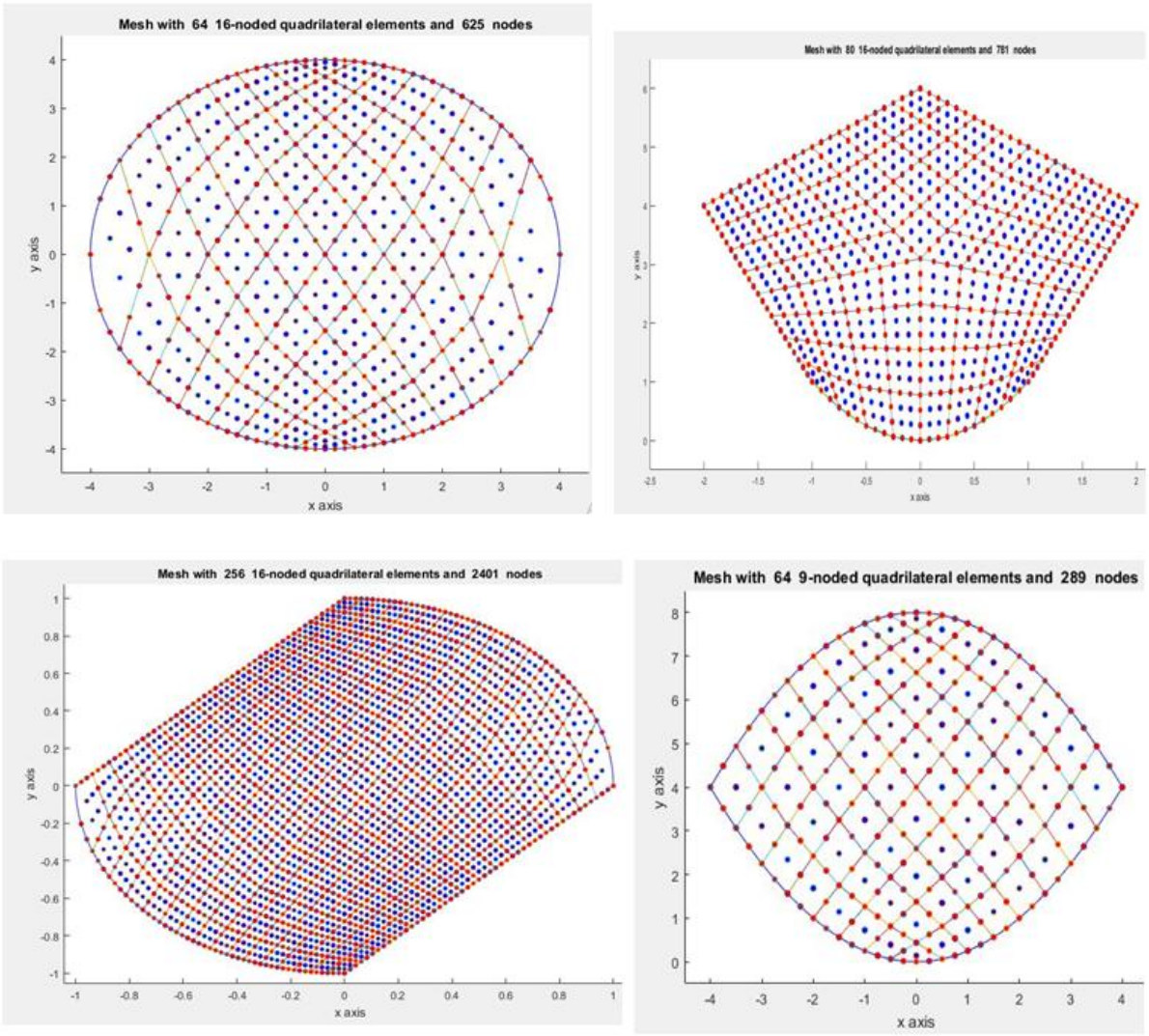


Figure – 7: p-version meshes of convex domains

III. Discussions

The time demand and the necessity to develop a new, more straightforward technique of generating all quadrilateral meshes are entirely undertaken in this study. We tried to develop the algorithm for the said task to utilize all the useful features of MATLAB. Finally, we have developed an algorithm and a computer code in MATLAB to generate both h- and p-version quadrilateral element meshes of any arbitrary convex domains according to the user's choice. The developed code uses a side-based mesh generation technique where only the coordinates of the vertices are needed to provide. Then other internal nodes of the elements are duly calculated, and as a result, it provides all the necessary data and displays the desired mesh of the domain under consideration. If the convex domain contains one or more curved sides, the functions that describe those sides should be provided along with the coordinates of vertices of that portion. In many test cases, the code is tested and found very lovely meshes. For clarity and reference, we have included a few convex domains and their meshes with different types of quadrilaterals.It is anticipated that such domains and corresponding meshes will be sufficient for a clear understanding of the suitability and efficiency of the developed algorithm and MATLAB code of this study.

IV. Conclusion

We considered an automatic mesh generation scheme for an arbitrary convex domain constituted by straight lines or curves employing lower or higher-order quadrilateral finite elements. For this, we developed first the general algorithm for h- and p- version meshes with quadrilaterals. The algorithm requires only the information of sides, the choice of the order along with the type of elements. It also allows one to form the desired fine mesh byproviding the number of refinements. Secondly, we developed the computer code in MATLAB based on the algorithm that provides all the valuable and needful outputs of the nodal coordinates,

relation between local and global nodes of the elements, and displays the desired meshes. For the experimental interest, we considered some convex domains of which the boundary constituted (1) only with straight sides, (2) by straight and curved sides, and (3) only with curved sides. The suitability and efficiency of the scheme substantiated through the demonstration of several test cases of mesh generation. We strongly assert that the code for automatic h- and p- version mesh generation employing the quadrilateral elements will find immense application in the FEM solution procedure.

References

[1]. Karim, M.S.(2000) Integration of some Bivariate Polynomials with Rational Denominators- An Application to Finite Element Method. A PhD Thesis, Dept. of Math., Bangalore University, Bangalore-560001, India.

[2]. https://en.wikipedia.org/wiki/Finite_element_method

[3]. Szabó, B. A. and Mehta, A. K., "p-Convergent Finite Element Approximations in Fracture Mechanics." International Journal for Numerical Methods in Engineering 12, pp. 551-560, 1978

[4]. Dhainaut, M. (1997), A Comparison Between Serendipity and Lagrange Plate Elements in The Finite Element Method. Commun. Numer. Meth. Engng., 13: 343-353. DOI:10.1002/(SICI)1099-0887(199705)13:5<343::AID-CNM60>3.0.CO;2-2

[5]. Bar–Yoseph, P. (1981), A comparison of various finite elements schemes for the solution of the Navier-Stokes equations in rotating flow. International Conference on Finite Elements in Flow Problems, 3rd, Banff, Alberta, Canada, June 10-13, 1980, Proceedings. Volume 1. (A82-17651 06-34) Calgary, Alberta, Canada, University of Calgary, 1981, p. 132-142.

[6]. Zienkiewicz, O. C. and Phillips, D. V. (1971), An automatic mesh generation scheme for plane and curved surfaces by 'isoparametric' coordinates. Int. J. Numer. Meth. Engng., 3: 519-528. DOI:10.1002/nme.1620030407

[7]. Imafuku, I., Kodera, Y. , Sayawaki, M. and Kono, M. (1980), A generalized automatic mesh generation scheme for finite element method. Int. J. Numer. Meth. Engng., 15: 713-731. DOI:10.1002/nme.1620150508

[8]. Lo, S. H. (1985), A new mesh generation scheme for arbitrary planar domains. Int. J. Numer. Meth. Engng., 21: 1403-1426. DOI:10.1002/nme.1620210805

[9]. Zhu, J. Z., Zienkiewicz, O. C., Hinton, E. and Wu, J. (1991), A new approach to the development of automatic quadrilateral mesh generation. Int. J. Numer. Meth. Engng., 32: 849-866. DOI:10.1002/nme.1620320411

[10]. Rathod, H.T., Rathod, B. (2018), A New Approach to Automesh Generation of graded triangularand quadrilateral Finite Elements over Analytical Surfaces by using theParabolic Arcs passing through four points on the Boundary Curve, International Journal of Engineering and Computer Science, (IJECS), vol. 7, issue 9, pp. 24214-24310. DOI: 10.18535/ijecs/v7i9.03

[11]. Rathod, H.T., Devi, K.Sugantha (2016), A New Approach To Automatic Generation Of All Quadrilateral Meshes Over A Linear Convex Polygon With H-Refinements For Finite Element Analysis, Int. J. Engg. And Computer Science. (IJECS), vol. 5, issue 7, 17172-17238. DOI: 10.18535/ijecs/v5i7.10

[12]. Alan Edaman (2004), Mathematics 18.337, Computer Science 6.338, SMA 5505, Applied parallel computing, pp. 167-173. https://courses.csail.mit.edu/18.337/2006/book/Lectures_2004.pdf

Appendix A

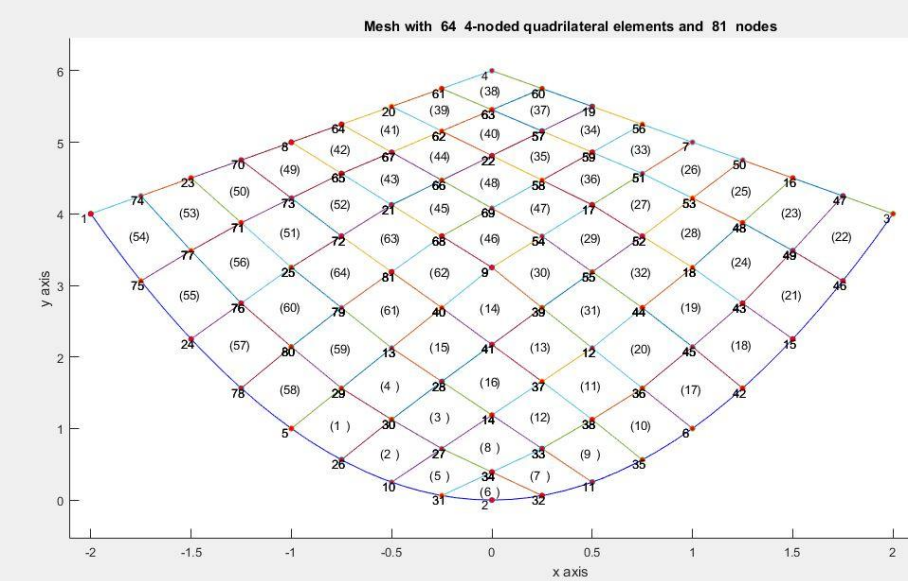
In this section, we presentsample MATLAB commands to get meshes of various convex domains. The meshed domains generated through the execution of the codes are also presented.

1. Meshing a convex domain with one curved side (of function $y = x^2$)

```
>> Assembled_Program_for_curvature_sided_polygonal_domain
Number of vertices? 4
Instruction: 1) Enter the nodal co-ordinates in an ascending order/descending order

e.g. [-1 0] -> [0 0] -> [1 0] or in reverse

Enter the co-ordinates:
Node 1: [-2 4]
Node 2: [0 0]
Node 3: [2 4]
Node 4: [0 6]
Enter order of element(1 or 2 or 3):1
Type of Element(LAGRANGE = 1, SERENDIPITY = 2)    1
Does the domain have curve side/sides? Yes: Not all sides curved = 1, All sides Curved = 2, No = 0;    1
Type of curvatures:
  curve of same function = 1, curve of different function = 2;       1
How many curved sides are there? 1
Refinement No. :  3
Enter function of the curvature:  @(x) x.^2
Enter boundary nodes which are in the curvature part: [1 2 3]
```



2. Meshing a convex domain enclosed two curves (function:$y_1 = \frac{x^2}{4}, y_2 = -\frac{x^2}{4} + 8$)

```
>> Assembled_Program_for_curvature_sided_polygonal_domain
Number of vertices? 4
Instruction: 1) Enter the nodal co-ordinates in an ascending order/descending order

e.g. [-1 0] -> [0 0] -> [1 0] or in reverse

Enter the co-ordinates:
Node 1: [-4 4]
Node 2: [0 0]
Node 3: [4 4]
Node 4: [0 8]
Enter order of element(1 or 2 or 3):2
Type of Element(LAGRANGE = 1, SERENDIPITY = 2)     1
Does the domain have curve side/sides? Yes: Not all sides curved = 1, All sides Curved = 2, No = 0;     2
Refinement No. :  4
Expression of curve 1: @(x) (x.^2)/4
Expression of curve 2: @(x) (x.^2)/4
Expression of curve 3: @(x) -(x.^2)/4+8
Expression of curve 4: @(x) -(x.^2)/4+8
End points of each curve boundary(minimum 3 points):    [1 2;2 3;3 4;4 1]
```