A MAXIMUM INSCRIBED CIRCLE ALGORITHM BASED ON VORONOI DIAGRAMS AND GEOMETRY EQUATIONS

Burak Beyhan[1], Cüneyt Güler[2], Hidayet Tağa[3]

[1] Assoc. Prof. Dr. Burak Beyhan, Mersin University, Department of City and Regional Planning
Mersin Üniversitesi, Mimarlık Fakültesi, Çiftlikköy Kampüsü, Yenişehir / Mersin Turkey
phone: +90-324-3610001 (17739) fax: +90-324- 3610109 e-mail: burakbeyhan@mersin.edu.tr

[2] Prof. Dr. Cüneyt Güler, Mersin University, Department of Geological Engineering
Mersin Üniversitesi, Mühendislik Fakültesi, Çiftlikköy Kampüsü, Yenişehir / Mersin Turkey
phone: +90-324-3610001 (17314) e-mail: cguler@mersin.edu.tr

[3] Assist. Prof. Dr. Hidayet Tağa, Mersin University, Department of Geological Engineering
Mersin Üniversitesi, Mühendislik Fakültesi, Çiftlikköy Kampüsü, Yenişehir / Mersin Turkey;
phone: +90-324-3610001 (17326) e-mail: htaga@mersin.edu.tr

Abstract
The aim of this study is to develop an algorithm for the calculation of Maximum Inscribed Circle (MIC) that can be placed within a polygon feature (PF) by using Free and Open Source Software (FOSS) for GIS. Compared with other parameters developed for PFs, there is no simple algorithm for the computation of MIC for different PFs. The algorithm developed in this study for the computation of MIC is based on the Voronoi diagrams and analytical geometry equations. The algorithm developed for the approximation of MIC can be applied to both convex and concave PFs. For the implementation of the algorithm, Eclipse IDE platform and OpenJUMP libraries written in Java is used. What is evident from the various runs of the script produced on the base of the algorithm for a set of regular and irregular PFs is that it is successful in finding MIC.

Keywords: *Maximum Inscribed Circle, Algorithm, Vector Data, Free and Open Source Software for GIS*

INTRODUCTION

The aim of this paper is to develop an algorithm for the calculation of MIC that can be placed within a polygon feature in vector data format by using FOSS for GIS. MIC is used in a wide range of fields, ranging from cartography (Shen et al., 2015; Sun, 2016), planning (Walz and Schumacher, 2005; Li, Goodchild, and Church, 2013; Brezina, Graser, and Leth, 2017) and geology (Powers, 1953) to biology (Tsygankov et al., 2014), military-security (Cheng, Li, and Zhu, 2012), and engineering applications (Meng et al., 2011; Liu et al. 2016, 2016; Li and Shi, 2009). Compared with other parameters developed for polygon features (PF), there is no simple algorithm for the calculation of MIC for different PFs. As it is suggested in various studies (Petrík et al., 2009; Shen et al., 2015; Sun, 2016), the Voronoi diagrams can be used for the calculation of MIC. Parallel to these suggestions in the algorithm developed in this study, Voronoi diagrams are effectively used. If it is assumed that MIC should be tangent to at least three edges of a PF (Sun, 2016) (though under certain conditions it can be tangent to only two edges), it follows that analytical geometry equations can also be effectively used for the calculation of MIC. Within this context, the algorithm developed for this study is composed of a number of sub-components;

1. Extraction of the points that are candidates for being center of MIC by using Voronoi diagrams,
2. Calculation of the first candidate point that can be used as the center of MIC,
3. Detection of the centers of the cores that may be used for the approximation of the center of MIC,
4. Processing of each core in order to approximate MIC of PF by using analytical geometry equations created on the base of the candidate points, and
5. Selection of the best circle calculated by using all possible cores and stages of analytical geometry equations in order to approximate MIC.

Each of these sub-components is elaborated in the subsequent parts of the paper in separate sections. After elaboration of these components and discussion of the results of the running the script created in OpenJUMP (2018), a FOSS for GIS, to implement the algorithm in comparison with other software alternatives, we draw on some concluding remarks.

EXTRACTION OF THE POINTS HAVING POTENTIAL FOR BEING THE CENTER OF MIC

For the extraction of the points that are candidates for being the center of MIC, Voronoi diagrams are created for the set of points located along PF subject to the calculation. For this purpose, in the algorithm (Figure 1) firstly, vertices of PF are extracted and stored in an array list (*as*) by creating mid-points and, if necessary, extra points between the couple of points forming the edges of PF.

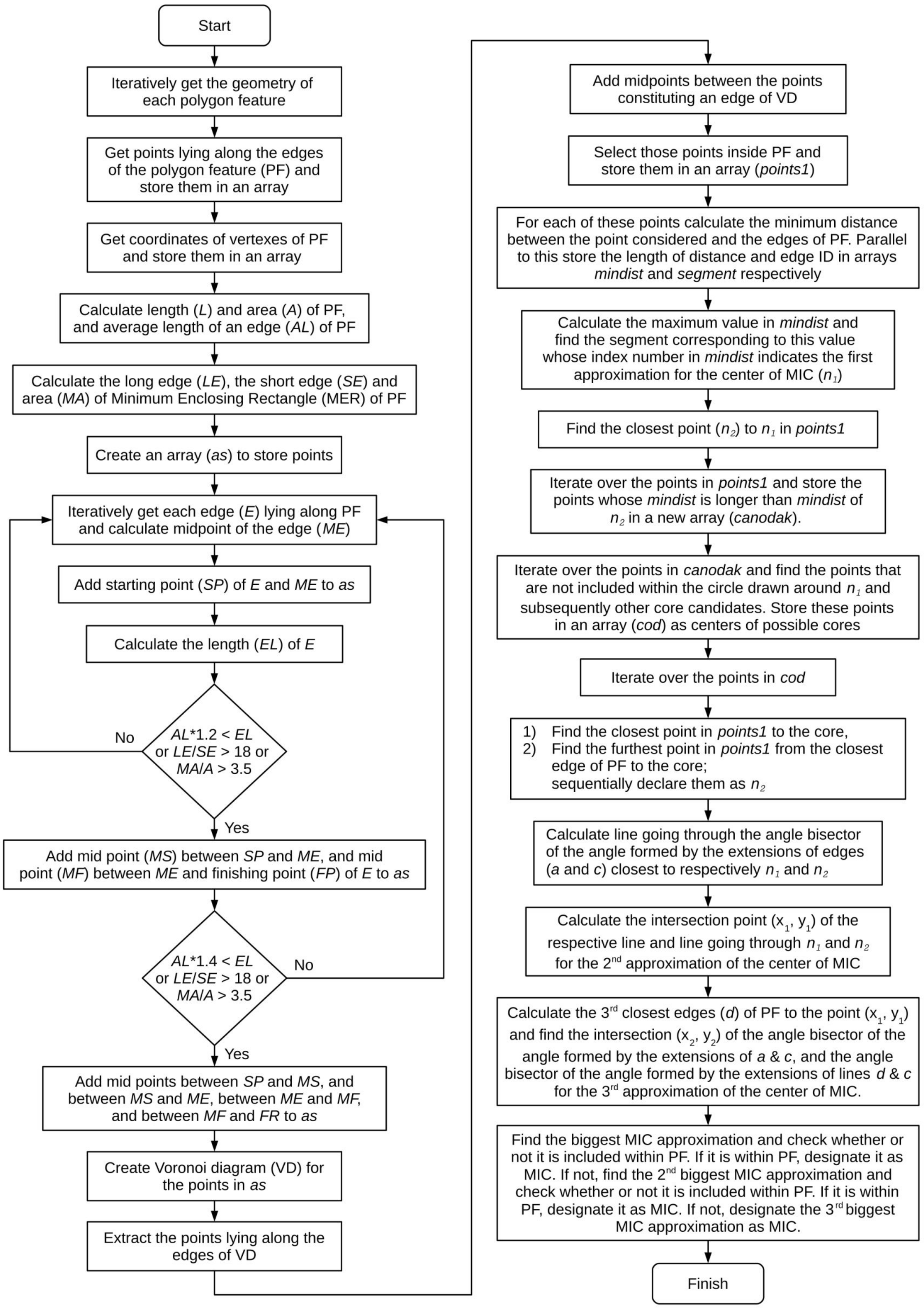
Start
Iteratively get the geometry of each polygon feature
Get points lying along the edges of the polygon feature (PF) and store them in an array
Get coordinates of vertexes of PF and store them in an array
Calculate length (L) and area (A) of PF, and average length of an edge (AL) of PF
Calculate the long edge (LE), the short edge (SE) and area (MA) of Minimum Enclosing Rectangle (MER) of PF
Create an array (as) to store points
Iteratively get each edge (E) lying along PF and calculate midpoint of the edge (ME)
Add starting point (SP) of E and ME to as
Calculate the length (EL) of E
AL*1.2 < EL or LE/SE > 18 or MA/A > 3.5
No
Yes
Add mid point (MS) between SP and ME, and mid point (MF) between ME and finishing point (FP) of E to as
AL*1.4 < EL or LE/SE > 18 or MA/A > 3.5
No
Yes
Add mid points between SP and MS, and between MS and ME, between ME and MF, and between MF and FR to as
Create Voronoi diagram (VD) for the points in as
Extract the points lying along the edges of VD
Add midpoints between the points constituting an edge of VD
Select those points inside PF and store them in an array (points1)
For each of these points calculate the minimum distance between the point considered and the edges of PF. Parallel to this store the length of distance and edge ID in arrays mindist and segment respectively
Calculate the maximum value in mindist and find the segment corresponding to this value whose index number in mindist indicates the first approximation for the center of MIC (n1)
Find the closest point (n2) to n1 in points1
Iterate over the points in points1 and store the points whose mindist is longer than mindist of n2 in a new array (canodak).
Iterate over the points in canodak and find the points that are not included within the circle drawn around n1 and subsequently other core candidates. Store these points in an array (cod) as centers of possible cores
Iterate over the points in cod
1) Find the closest point in points1 to the core,
2) Find the furthest point in points1 from the closest edge of PF to the core; sequentially declare them as n2
Calculate line going through the angle bisector of the angle formed by the extensions of edges (a and c) closest to respectively n1 and n2
Calculate the intersection point (x1, y1) of the respective line and line going through n1 and n2 for the 2nd approximation of the center of MIC
Calculate the 3rd closest edges (d) of PF to the point (x1, y1) and find the intersection (x2, y2) of the angle bisector of the angle formed by the extensions of a & c, and the angle bisector of the angle formed by the extensions of lines d & c for the 3rd approximation of the center of MIC.
Find the biggest MIC approximation and check whether or not it is included within PF. If it is within PF, designate it as MIC. If not, find the 2nd biggest MIC approximation and check whether or not it is included within PF. If it is within PF, designate it as MIC. If not, designate the 3rd biggest MIC approximation as MIC.
Finish

Figure 1. MIC algorithm developed in the study.

Placement of extra points between the couple of points is based on a set of conditions. The first condition is related to the average length of the edges of PF. If it is above a pre-defined level, extra points are placed along the edge concerned. In addition to this condition, the ratio of the width of the Minimum Enclosing Rectangle (MER) to the height of the MER and the ratio of the area of the MER to the area of PF are also used in order to place extra points along the edges of PF if they are above the predefined levels (18 for the former and 3.5 for the latter).

After creation of *as*, for the construction of Voronoi edges, existing software libraries in Java Topology Suite (JTS) of OpenJUMP are used (more particularly VoronoiDiagramBuilder). As each point located along the medial axis of a PF can actually be considered as a canditate for the center of MIC, in the algorithm the points (*points1*) that are both located along the Voronoi edges and covered by PF are considered as candidates for the center of MIC.

FINDING THE FIRST CANDIDATE POINT THAT CAN BE TAKEN AS THE CENTER OF MIC

Finding the first candidate point that can be taken as the center of MIC is actually a max–min problem of maximizing the minimum distance from an unknown point located along the medial axis of PF to all the edges of PF. Thus, in the subsequent part of the algorithm (Figure 1), firstly, the minimum distance (*R*) between each point in *points1* and the edges of the polygon feature is calculated and stored in another array (*mindist*). Parallel to this calculation, the index number for the edges closest to the points in *points1* is also stored in another array (*segment1*) for further calculations that will required to find the center of MIC. Consequently the index number for the point (n_1) having the maximum value in *mindist* is revealed by iterating over *mindist* in order to approximate the center (*index*) of MIC (Figure 2 and Figure 3). In this process, the objective function used to find the center of best MIC candidate (n_1) maximizes the radius (*R*) of circle tangent interior to one of the edges of PF (1):

$$F(x, y) = Max\{Min(R)\} \quad (1)$$

Parallel to this, the index information (n_1) for the edge closest to the point represented by *index* is also calculated. At this stage, the point represented by *index* is actually a good candidate for the center of MIC.

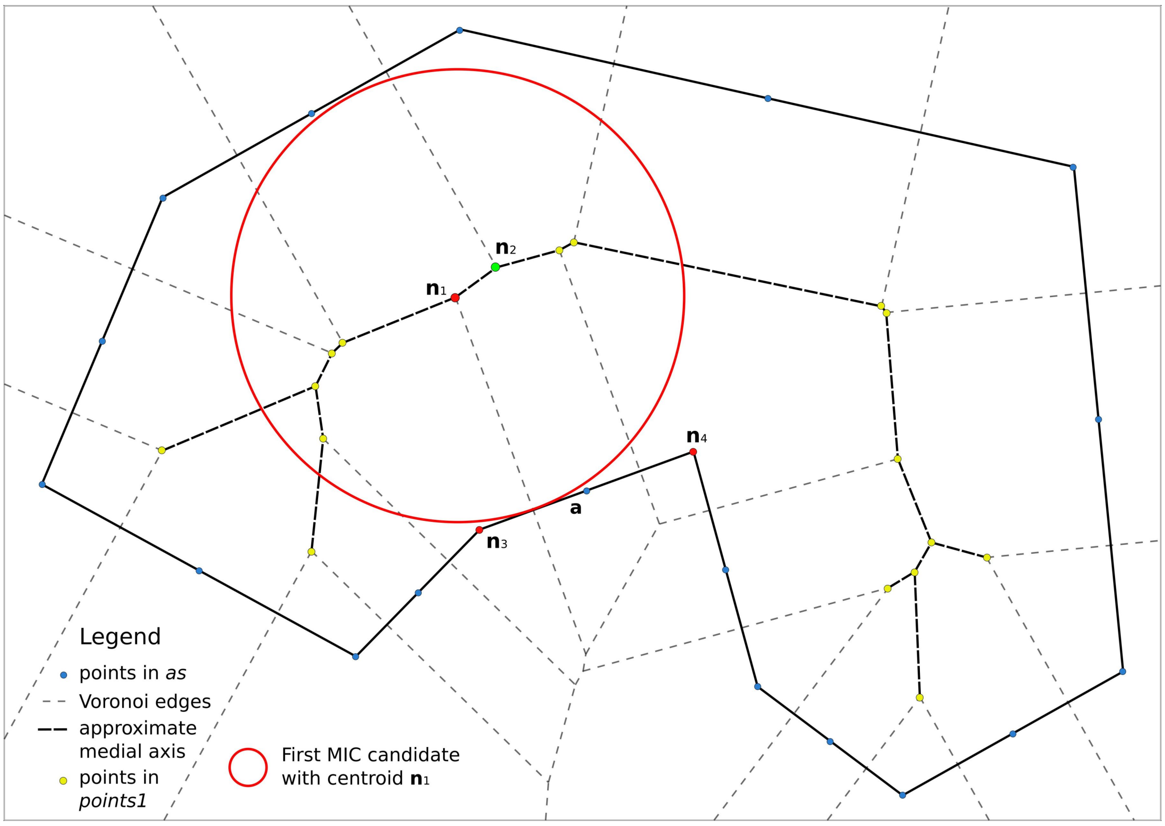

Figure 2. Sample polygon and first candidate for MIC (n_1).

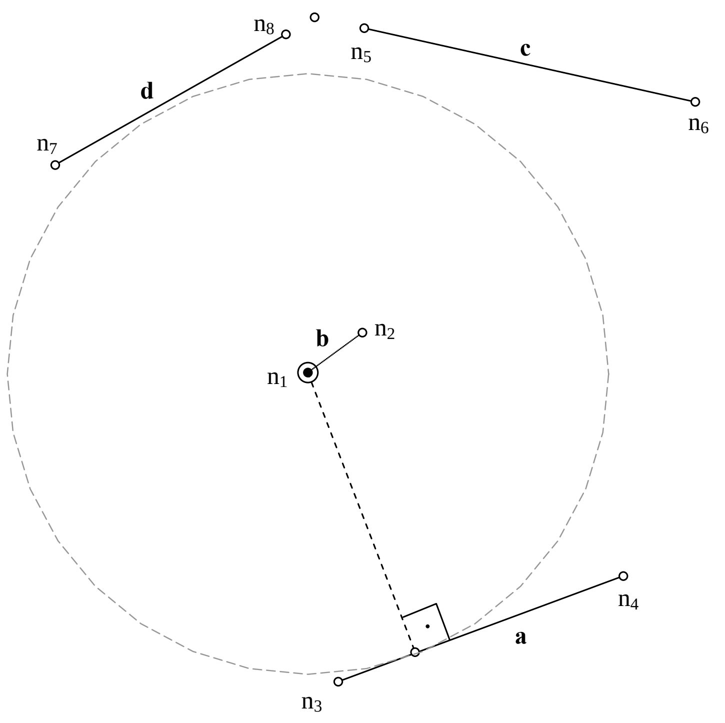

Figure 3. The first approximation for the center of MIC based on the points located along the medial axis.

UNCOVERING THE POTENTIAL CORES THAT MAY COVER THE CENTER OF MIC

Depending on the shape of PF there may be more than one core that should be used in the approximation of the center of MIC (Figure 4). Thus, in the algorithm all cores lying along the medial axis are considered in the calculation of MIC.

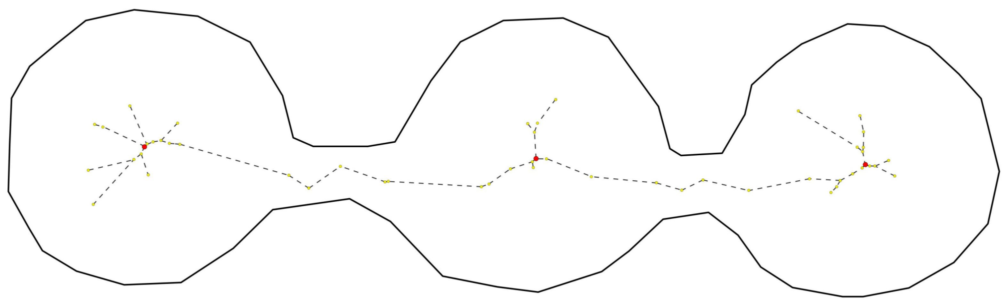

Figure 4. An illustrative example for multiple core problem (a polygon feature having three remarkable cores).

As it can be followed from the algorithm (Figure 1), for this purpose, firstly the closest point (n_2) in *points1* (yellow points in Figure 4 along medial axis) to n_1 is found. The objective function for the respective point can be given as below:

$$F(x,y) = Min(d_{i,n_1}) \quad (2)$$

where d_{i,n_1} stands for the distance between the point concerned in *points1* and n_1.

And then all the points in *points1* is checked against whether the distance between them and the edge closest to them (R_{od_i}) is more than the distance between n_2 and the edge closest to it (R_{n_2}). The respective set of points fulfilling this condition are stored in an array (*canodak*) and considered to have the potential to represent the centers of the cores. The points in *canodak* are further processed in order to find the centers of the cores. For this purpose, we iterate over the points (*od*) in *canodak* and find the points that are not included within the circle drawn around n_1 and subsequently other core candidates. These points are stored in a new array (*cod*) representing centers of possible cores:

$$cod_{\{x,y\}} = \bigcup_{i=1}^{s} od_i \text{ for } R_{od_i} > R_{n_2} \quad (2)$$

where *s* stands for the total number of cores.

PROCESSING CANDIDATE POINTS IN ORDER TO APPROXIMATE BEST MIC

In addition to the employment of Voronoi diagrams and subsequently medial axis of PF, for each core, analytical geometry equations are also used for the approximation of the best center for MIC that must be tangent to at least three edges of PF because it is known that on a 2D plane three non-collinear points define a circle (though under certain conditions it can be tangent to only two edges). Thus, if the circle drawn around n_1 is tangent to only one of the edges of the polygon feature, it means that it may not be the maximum inscribed circle (see Figure 2 and Figure 3). Thus, further considerations are required to find the actual MIC that should be tangent to at least another edge of PF concerned.

Assume that in Figure 3, n_1 is the point identified by *index* and *a* is the closest edge of the polygon feature to n_1. If n_2 is the closest point to n_1 compared with the other points located along the medial axis and *c* is the closest edge of the polygon feature to n_2, a better candidate for the center of MIC should lie between n_1 and n_2, and the circle drawn around it (with a radius d_1) should be tangent to both *a* and *c* (Figure 5 and Figure 6). The condition for being tangent to both *a* and *c* reveals that the respective point (x_1, y_1) lies at the intersection of *b* and the angle bisector of the angle formed by the extensions of lines *a* and *c*.

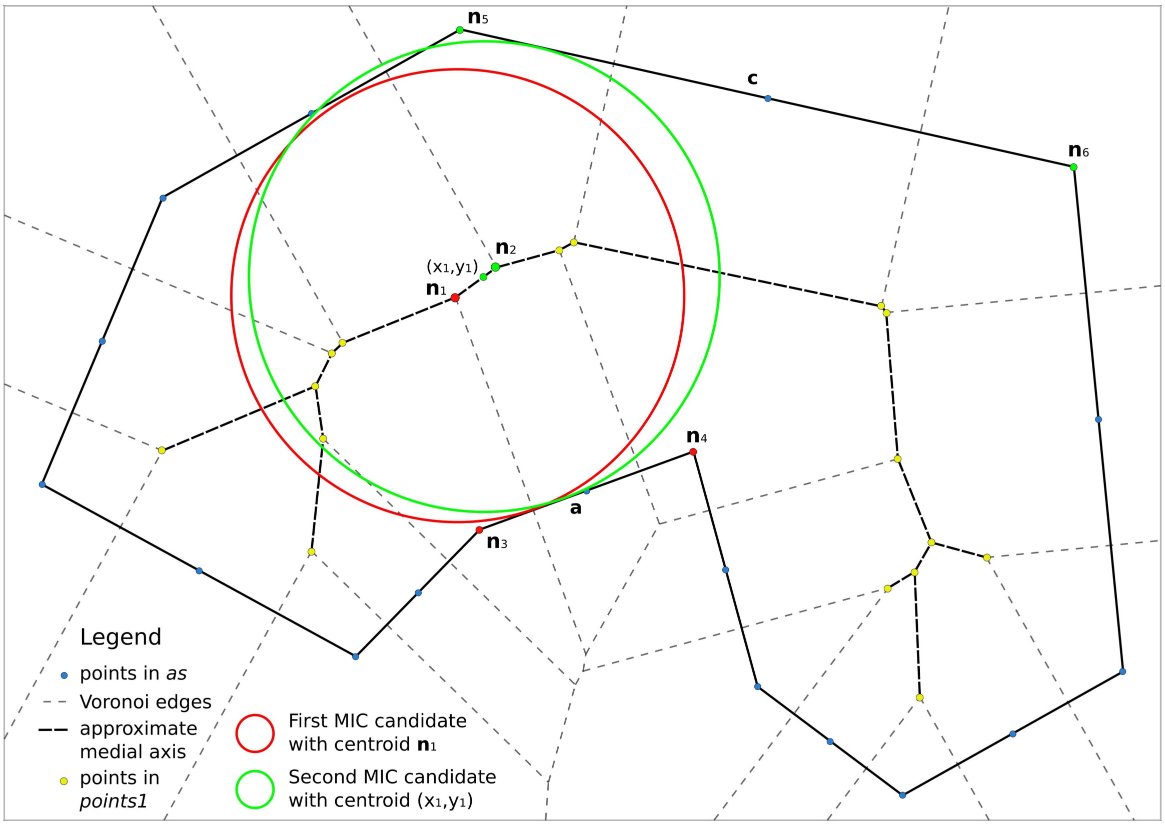

Figure 5. Sample polygon and second candidate for MIC (green circle drawn around the green point between n_1 & n_2).

By using the formula for a line, the intersection point (x_1, y_1) can be easily calculated. For this purpose, firstly, the index number for the point (n_2) located along the Voronoi edges and closest point to n_1 is calculated and stored in a variable (*indet1*). The index information (n_2) for the edge closest to the point represented by *indet1* is also calculated.

Subsequently, the intersection point of *a* and *c* (xi, yi) is calculated by constructing the required equation. After calculating the slope of the line going through the angle bisector of the angle formed by the extensions of lines *a* and *c*, the intersection point ((x_1, y_1) on Figure 6) of the respective line and line *b* is calculated for the second approximation of the center of MIC. The same calculation is also done for the furthest point in *points1* from the closest edge of PF to the core (Figure 1).

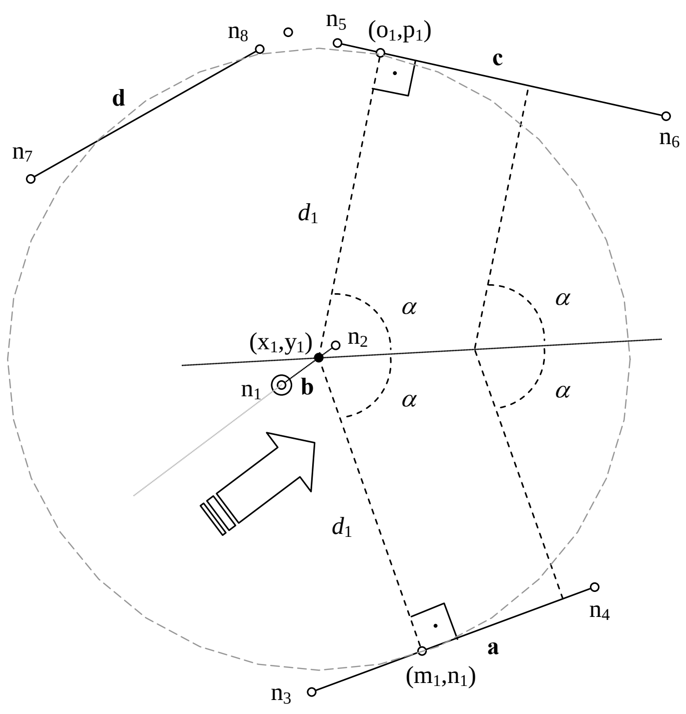

*Figure 6. The second approximation for the center of MIC based on the intersection of **b** and the angle bisector of the angle formed by the extensions of lines **a** and **c**.*

If the circle drawn around (x_1, y_1) is tangent to only two of the edges of the polygon feature, it may not be tangent to a third edge of PF and it may actually overflow the boundaries of the PFs. In other words, it means that it may not be the maximum inscribed circle (Figure 5 and Figure 6).

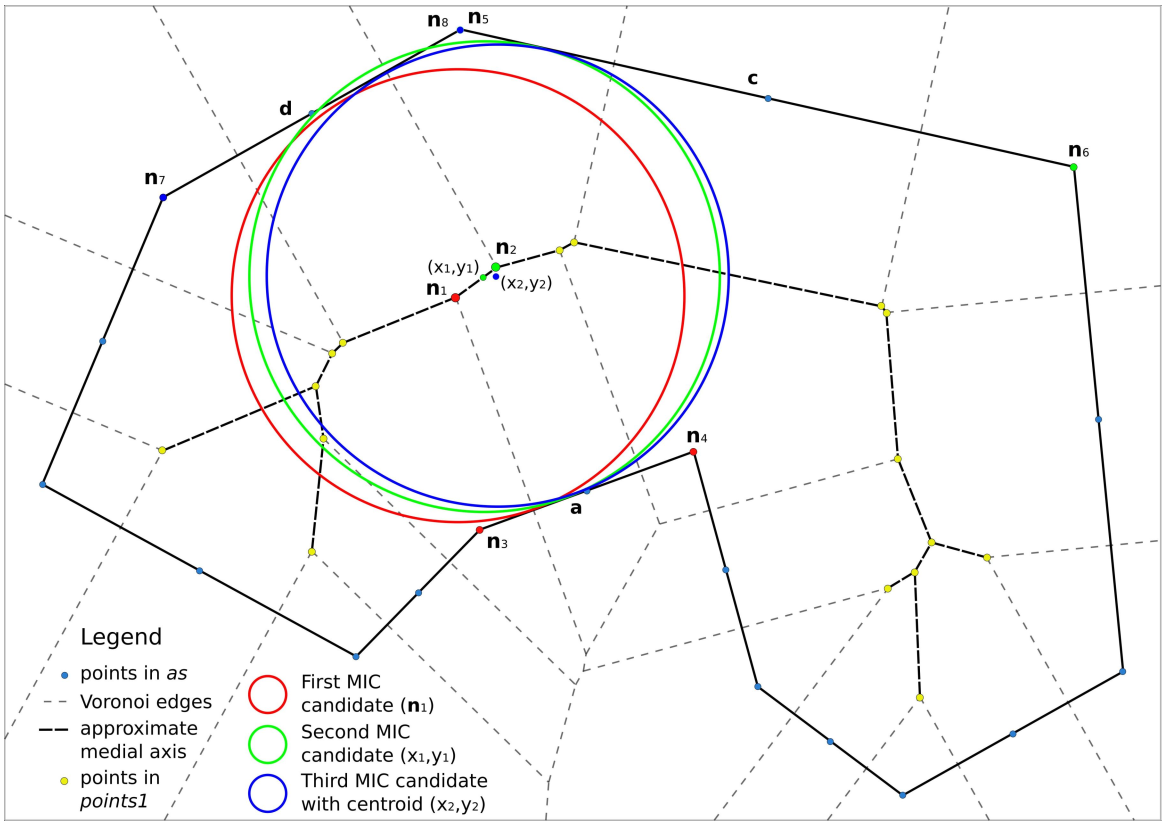

Figure 7. Sample polygon and third candidate for MIC (blue circle drawn around the small blue point).

Thus, further considerations are necessary to find the real MIC (Figure 7). Assume that in Figure 8, *a*, *c* and *d* is the first three closest edges of the polygon feature to the point (x_1, y_1). A better candidate (x_2, y_2) for the center of MIC should lie at the intersection of the angle bisector of the angle formed by the extensions of lines *a* and *c*, and the angle bisector of the angle formed by the extensions of lines *d* and *c* (Figure 8).

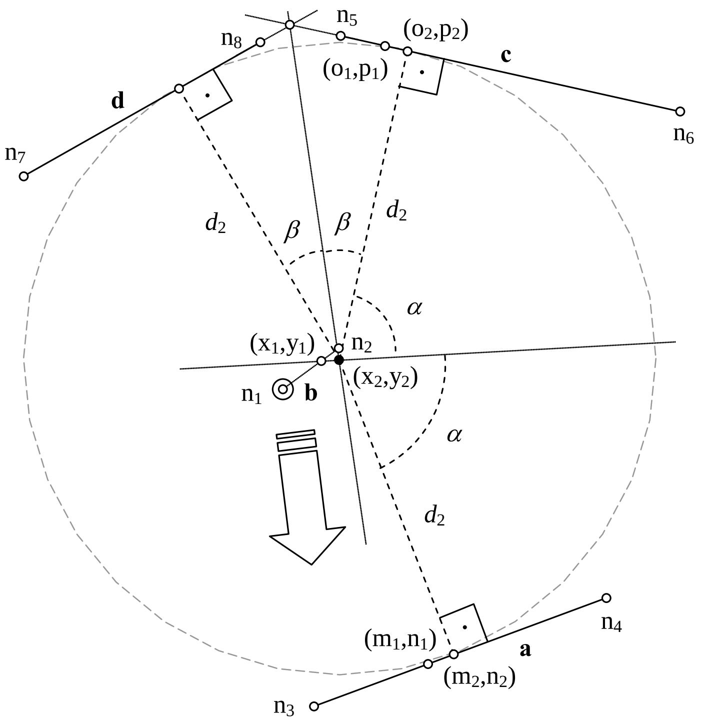

*Figure 8. The third approximation for the center of MIC based on the intersection of the angle bisector of the angle formed by the extensions of lines **a** and **c** with the angle bisector of the angle formed by the extensions of lines **d** and **c**.*

For concave polygons, given the fact that MIC may not be tangent to *a*, *c* or *d* because of the geometric properties of polygons concerned, some further consideration is actually required in the algorithm. Indeed, sometimes MIC may not be tangent to edges but touches their ending nodes. In this particular case, the line passing through the respective node and tangent to MIC should be taken into account in the calculations made for the approximation for the possible centers of MIC.

SELECTION OF BEST MIC ON THE BASE OF POTENTIAL CORES

In both the second and third approximation for the center of MIC, the resulting MIC should be checked against whether or not it is fully covered by the polygon feature and it is the biggest one compared with the others already calculated. This comparison should also be done for different cores subject to the analysis. The script produced for the calculation of MIC based on the algorithm developed in this paper takes all these considerations into account together with the fact that MIC may not be tangent to edges of PF but touches their ending nodes.

For the selection of the best MIC, built-functions (particularly spatial predicates such as “within”) available in JTS have been actively used. If all the MIC candidates are within PF, the one having the largest area is designated as MIC. Those candidates not completely covered by PF are omitted from the comparison operation. For example, the second candidate for MIC calculated for the sample polygon used in this study in Figure 7 is not fully within PF concerned. Thus, it is omitted from the comparison operation.

RESULTS OF THE RUNNING THE SCRIPT CREATED TO IMPLEMENT THE ALGORITHM IN COMPARISON WITH OTHER SOFTWARE ALTERNATIVES

In order to test the algorithm, the script created for the implementation of the algorithm has been run for a set of a regular (A) and irregular & complex (B) PFs in OpenJUMP (2018). Some of the results obtained after running the plugin can be seen in Figure 9. The actual spatial database used for irregular & complex PFs cover 491 PFs (mostly concave). Thus, only a limited set of them is given in Figure 9. Nevertheless, in Table 1 created for the comparison of performance of the script produced for the algorithm developed in this study with some other software programs and scripts having capability to calculate MIC, the statistics are based on the whole set of irregular & complex PFs. MIC function is called with various names in the respective software programs. In LayoutEditor (Thies, 2018) originally created in order to edit designs for MEMS (Micro-Electro-Mechanical Systems) and IC (Integrated Circuit) fabrication, it is called as "Biggest Inner Circle" (BIC) that is available under "Shape Utilities" menu item. In Flowmap (2013) produced for the analysis of spatial relationships, it is called as "Largest Inscribed Circle" (LIC) that is available under "Point in Polygon Analysis" menu item. In SAGA (2018), it can be calculated via "Largest Circles in Polygons" (LCP) tool available under "Tools Chains" menu item. Since the result of the respective tool involves more than one circle for a PF, the largest circle can be selected by comparing the all the circles calculated by the software with each other. In GeoWizard (2018) extension for ArcGIS Desktop, MIC is available as a tool under "Miscellaneous" menu item. The hardware configuration and operating system of the computer used for performance test is as following; a quad-core processor with 2.66 Ghz frequency and 16 GB of RAM on 64 bit Windows 10 operating system.

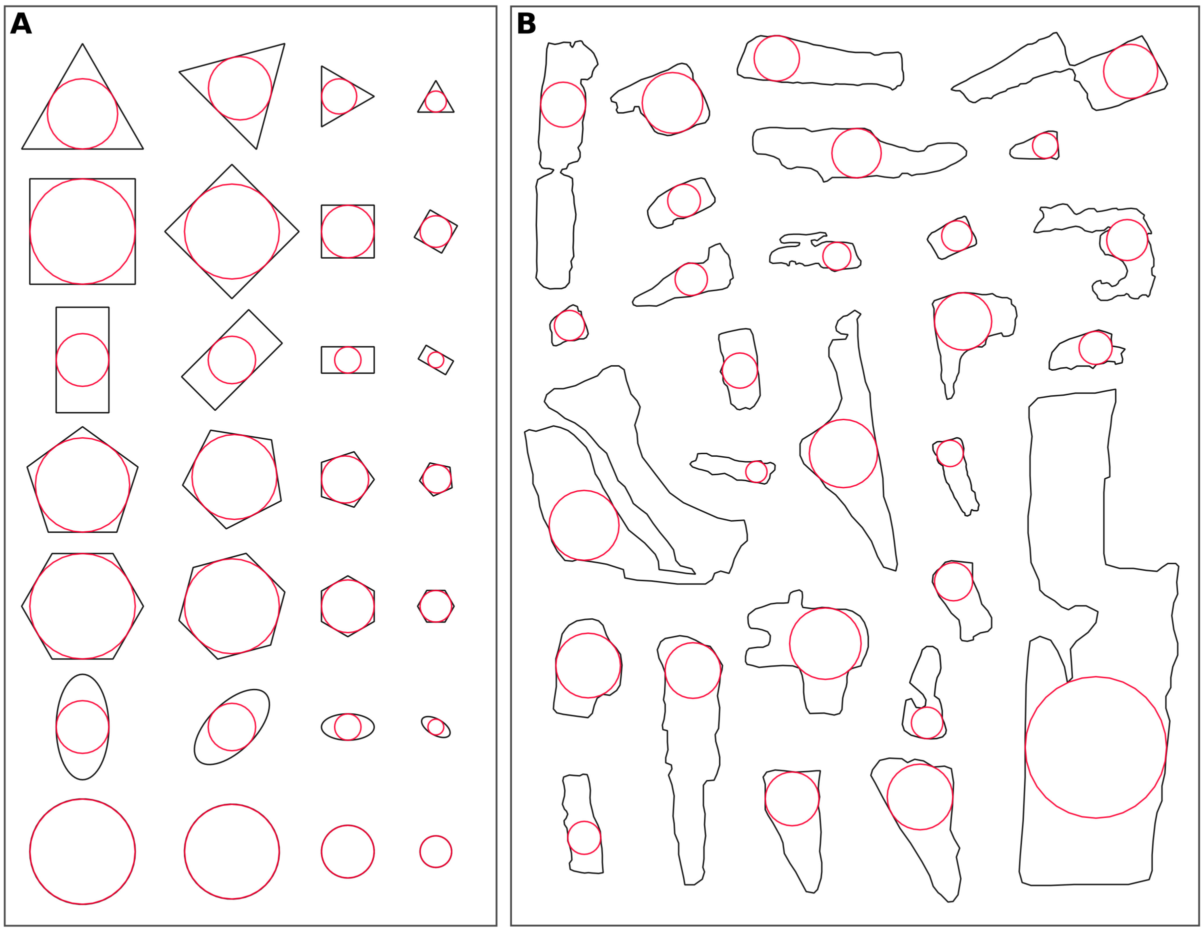

Figure 9. Various runs of the script for a set of regular (A) and irregular and complex (B) polygon features.

As it is evident from Table 1, MIC script produced for OpenJUPM and MIC tool available in ET GeoWizards are more successful in finding MICs compared with other software programs. For only 8 and 7 PFs, calculated MICs ignorably overflow or touch the boundaries of PFs concerned, respectively in OpenJUMP and ET GeoWizard. Overall for all the MICs calculated by using the respective plugins, more than 99.99% of area of the MICs concerned are contained within the boundaries of the PFs. Nevertheless, in the case of Flowmap, calculated MICs overflow the boundaries of the PFs concerned for more than 370 PFs even though the script was also run in "Slow & Accurate" mode (in Flowmap there are different options for the calculation of MIC; (1) Quick & Dirty, (2) Intermediate (default option), (3) Slow & Accurate, and (4) Super Slow & Perfect (almost)). In Flowmap, at most for only 33% of the PFs, more than 99.99% of area of the MICs concerned are contained within the boundaries of the PFs if it is run in Intermediate or Slow & Accurate modes. LCP tool in SAGA also requires a resolution value to be specified. For higher values, no result is obtained for small PFs. However when it is set to 1 meter, it takes more than 10 hours to obtain the results. Compared with the other software programs, BIC in LayoutEditor has the best performance in speed. The accuracy of MICs calculated by LayoutEditor is also satisfactory. Nevertheless, it could not properly read and write GIS data formats generally used in GIS software programs. In order to test the performance of the software concerned, files have been saved in SVG format. Other options (such as DXF) did not work in LayoutEditor. In order to compare the results of BIC utility in LayoutEditor with the results of other software scripts, the results concerned are rescaled to their original coordinates.

Table 1. Comparison of MIC script produced for OpenJUMP with some other software programs and scripts.

Compared Charachteristics	MIC script in OpenJUMP	BIC utility in LayoutEditor	MIC tool in ET GeoWizards	LCP tool in SAGA	LIC in FlowMap	
					Intermediate	Slow & Accurate
Number of MICs completely covered by their PFs	483	347	484	368	116	96
Share of PFs covering more than 99.99% of the area of their MIC	100	100	100	80.86	32.99	31.57
Number of MICs overflowing or touching the boundaries of their PFs	8	144	7	123	375	395
Speed of the script	10 seconds	2 seconds	48 seconds	12 hours	115 seconds	19 min.

For the comparison of performances of the MIC algorithm developed in this study and MIC tool in ET GeoWizard, another table is prepared. With this intention, for each PF, the radius of the MIC computed by the script created for this study (r_{micoj}) is compared with the one calculated by ET GeoWizard (r_{micet}) in order to obtain a comparison parameter (*comp*). Subsequently, if r_{micoj} is greater than r_{micet}, value of *comp* is set to $(r_{micoj} - r_{micet})*100/r_{micet}$. If r_{micoj} is smaller than r_{micet}, value of *comp* is set to $(r_{micet} - r_{micoj})*100/r_{micoj}$. Lastly, the number of PFs is counted according to the intervals defined for the value of *comp* and shown in Table 2.

Table 2. Comparison of MIC script run in OpenJUMP with ET GeoWizards' MIC tool.

Intervals for *comp*	Irregular & Complex - Concave PFs		Convex Regular PFs
	$r_{micoj} > r_{micet}$	$r_{micoj} < r_{micet}$	$r_{micoj} > r_{micet}$
$0 < comp < 0.5$	227	11	11
$0.5 \le comp < 1$	220	5	16
$1 \le comp < 2.5$	26	-	1
Total	473 (96.33%)	16 (3.67%)	28 (100.00%)

What is evident from Table 2 is that, for all of the convex regular PFs and 96.33% of the irregular & complex PFs, the MIC calculated by the script created for the algorithm developed in this study is larger than the one calculated by MIC tool in ET GeoWizard, which unveils that the MIC algorithm developed in this study is not only faster but also more successful in finding MIC compared with the alternative software.

CONCLUDING REMARKS

What is evident from the various runs of the script produced on the base of the algorithm developed in this study for a set of regular and irregular complex polygon features is that it is successful in finding MIC compared with the existing scripts and software programs produced for this purpose or having a functionality for the calculation of MIC. Particularly, compared with the other scripts and software programs integrated into a GIS environment or having capability to read and write geographic data, the algorithm created for the calculation of MIC in this study is both faster and more accurate. Thanks to the existing libraries in FOSS for GIS, the algorithm developed in this study could easily be implemented in OpenJUMP as a plugin by benefiting the libraries concerned (such as the ones for the creation of Voronoi diagrams and spatial predicates in JTS). Without the availability of the libraries created by FOSS community, it would not be possible for this study to solely focus on the perfection of the algorithm developed to find MICs of PFs.

The algorithm developed in this study will be available as a plugin in OpenJUMP. Yet, some further adjustments are required in n_2 parameter in order to better approximate the center of MIC. This is designated as one of the topics that can be addressed by the future studies aiming at finding MIC. Development of morphometric parameters for PFs is another important topic for future research trying to understand the nature of spatial configurations at different scales.

REFERENCES

Brezina, T., Graser, A., & Leth, U. (2017) Geometric methods for estimating representative sidewalk widths applied to Vienna's streetscape surfaces database. *Journal of Geographical Systems,* 19(2), 157-174.

Cheng, T., Li, P., & Zhu, S. (2012) An algorithm for jammer localization in wireless sensor networks. In *Advanced Information Networking and Applications, 2012 IEEE 26th International Conference* (Fukuoka, Japan, 2012) 724-731.

ET GeoWizards (2018) *ET GeoWizards Ver. 11.5.* Available online at (10.02.2018) https://www.ian-ko.com/ETGeoWizards.html

Flowmap (2013) *Educational Flowmap Release 7.4.* Available online at (20.11.2017) http://flowmap.geo.uu.nl/

Thies, J. (2018) *LayoutEditor Free Version*. Available online at (10.01.2018) http://www.layouteditor.net/

Li, W., Goodchild, M.F., & Church, R. (2013) An efficient measure of compactness for two-dimensional shapes and its application in regionalization problems. *International Journal of Geographical Information Science,* 27(6), 1227-1250.

Li, X. & Shi, Z. (2009) The relationship between the minimum zone circle and the maximum inscribed circle and the minimum circumscribed circle. *Precision Engineering,* 33(3), 284-290.

Liu, F., Xu, G., Liang, L., Zhang, Q., & Liu, D. (2016) Minimum Circumscribed Circle and Maximum Inscribed Circle of Roundness Deviation Evaluation with Intersecting Chord Method. *IEEE Transactions on Instrumentation and Measurement,* 65(12), 2787-2796.

Meng, F., Chunguang, X., Haiming, L., Hao, J., & Xiao, D. (2011) A Quick Algorithm of Maximum Inscribed Circle Method for Roundness Evaluation. In *International Conference on System Science, Engineering Design and Manufacturing Informatization* 348-351.

OpenJUMP (2017) OpenJUMP Version 1.10. Available online at (21.08.2017) http://www.openjump.org/

Petrík, M., Kováč, J., Kaťuch, P., Bednarčíková, L., Hudák, R., & Živčák, J. (2009) Roundness: Determining the Reference Circle for MCCI and MICI System. In *Proceedings of the 7th International Conference Measurement* (Smolenice, Slovakia, 2009) 352-355.

Powers, M.C. (1953) A new roundness scale for sedimentary particles. *Journal of Sedimentary Research,* 23(2), 117-119.

SAGA (2018) *SAGA Version: 6.3.0.* Available online at (25.02.2018) http://www.saga-gis.org/

Sun, S. (2016) Symbolize map distortion with inscribed circles in polygons. *International Journal of Cartography,* 2(2), 166-185.

Shen, Z., Yu, X., Sheng, Y., Li, J., & Luo, J. (2015) A Fast Algorithm to Estimate the Deepest Points of Lakes for Regional Lake Registration. *PLoS ONE*, 10(12): e0144700. doi:10.1371/journal.pone.0144700

Tsygankov, D., Bilancia, C.G., Vitriol, E.A., Hahn, K.M., Peifer, M., & Elston, T.C. (2014) CellGeo: a computational platform for the analysis of shape changes in cells with complex geometries. *J Cell Biol,* 204(3), 443-460.

Walz, U. & Schumacher, U. (2005) Landscape fragmentation in the Free State of Saxony and the surrounding border areas. In *Networking Environmental Information. Proceedings of the 19th International Conference on Informatics for Environmental Protection* (Brno, Czech Republic) 754-758.

BIOGRAPHY

Burak Beyhan is an Associate Professor at the Department of City and Regional Planning, Mersin University. He received his degrees (Bachelor of City Planning - B.CP., Master of Regional Planning - M.RP., and Doctor of Philosophy - Ph.D.) in the Department of City and Regional Planning at Middle East Technical University, in Ankara, Turkey. His main research interests are in the areas of urban and regional planning, regional development and innovation systems, geographic information systems (GIS) in planning, and planning history in Turkey.

Cüneyt Güler is a Professor at the Department of Geological Engineering, Mersin University. He received his Bachelor of Science (B.Sc.) in the Department of Geological Engineering, İstanbul Technical University, in İstanbul, Turkey, and received his Master of Science (M.Sc.) and Doctor of Philosophy (Ph.D.) in the Department of Geological Engineering, Colorado School of Mines, Colorado, USA. His main research interests are in the areas of geological engineering, GIS and hydrogeology.

Hidayet Tağa is an Assistant Professor at the Department of Geological Engineering, Mersin University. He received his degrees (B.Sc., M.Sc. and Ph.D.) in the Department of Geological Engineering, Çukurova University, in Adana, Turkey. His main research interests are in the areas of geological engineering, GIS and land-use planning.