FREEFORM-BASED CONSTRUCTION SITE LAYOUT OPTIMIZATION

Ibrahim Abotaleb[1], Khaled Nassar[2], Ossama Hosny[3]
[1] *Graduate Student, Department of Construction & Architectural Engineering, The American University in Cairo, Egypt.*
[2] *Associate Professor, Department of Construction & Architectural Engineering, The American University in Cairo, Egypt.*
[3] *Professor, Department of Construction & Architectural Engineering, The American University in Cairo, Egypt.*

Abstract:

Traditional approaches to the construction site layout problem have been focused mainly on rectilinear facilities where the site facilities are modeled as either a set of discrete cells using an orthogonal grid, equal and unequal rectangular shapes, or circular geometries. This is a fair abstraction of the problem; however it ignores the fact that many facilities on the construction sites assume non-rectilinear shapes that allow for better compaction within tight sites. The main focus of this research is to develop a practical real-life construction site layout tool that uses freeform shapes for the facilities, specifically splines, and introduces different proximity importance measures. The freeform nature of the construction site facilities allows for better compaction of these facilities in restricted sites. In this research, a new "dynamic" model is developed where the facilities dimensions and shapes are automatically adjusted to fit into strict spaces. The site layout modeling was formulated on commercial parametric modeling tools (Rhino® and Grasshopper®) and the optimization was performed through genetic algorithms.

Keywords: Site layout, optimization, genetic algorithms, freeform.

1 Introduction:

Construction site layout planning involves identifying, sizing, and placing temporary facilities within the boundaries of construction site to fulfil certain objective functions such as minimizing travel time, minimizing safety hazards, or maximizing the proximity measures set by project managers. A detailed planning of the site layout and location of temporary facilities can enable the management to make considerable improvement by minimizing travel time, waiting time, and increasing worker morale by showing better and safer work environment. An accurate representation of the construction site is important for site layout modeling, as it enables the development of more realistic and efficient layouts.

The problem of site layout planning has been solved by researchers using mainly two techniques; heuristic techniques and mathematical optimization models to produce the optimum solutions. Heuristic methods such as the model presented by Tommelien et al. (1992) and Cheng & O'Connor (1996) produce good but not optimal solutions, and cannot be fairly adopted for large complex projects. Mathematical techniques are capable of solving more complex site layout planning problems with different types of variables and constraints. Mathematical techniques involve the identification of one or more goals that the site layout strives to achieve, the modeling and mathematical term for the end goal is the "objective function". The goal is to optimize the objective function by either maximizing or minimizing it through the alterations of the different model variables. Several mathematical techniques and models were developed to solve site layout problems. Most famous of them are linear programming, genetic algorithms (Li & Love, 1998; Hegazy & Elbeltagi, 1999; Zouein et al., 2002; Osman et al., 2003), artificial neural networks (Yeh, 1995), artificial bee colony (Yahya & Saka, 2014), artificial ant colony (Ning et al., 2011), fuzzy logic (Elbeltagi & Hegazy, 2001; Xu & Li, 2012), simulated annealing (Andayesh & Sadeghpour, 2014), and dynamic programming (Ning et al., 2011; Khalafallah & El-Rayes, 2002; El-Rayes & Said, 2009).

Research in the field of site layout planning is targeted on either developing the modeling techniques for the site facilities and obstacles, finding better optimization techniques to find optimum solutions, investigating better proximity measures, examining and comparing different objective functions, or investigating the effect of time and scheduling on the different site layout variables. This research is oriented towards finding better algorithms for modeling the site layout facilities and obstacles since the research in this area has not been widely developed as shown in Figure 1. Li & Love (1998) modeled the site facilities in predetermined locations; where the number of predetermined places should be equal to or greater than the number of predetermined facilities. So there is no flexibility in moving the site facilities, except in the pre-determined locations. The variable of each facility would just be its location number. In 1999, Hegazi & Elbeltagi presented a more flexible modeling for the site facilities by using a two-dimensional grid, where each facility is modeled as a number of grid units that add up to the facility area. Osman et al. (2003) provided a CAD-based model for site facilities, with almost the same concept as Hegazi & Elbeltagi's model (1999) by the dividing the site into perpendicular grids and enclosing the facilities inside the grids. In 2013, Andayesh & Sadeghpour represented site facilities by their minimum bounding circles to facilitate the optimization process. However, this representation also results in a waste of space. In 2014, Yahya & Saka presented a model that modeled facilities as rectangular geometries that allow for horizontal and vertical alignment with site boundaries that are represented as lines with specified slopes.

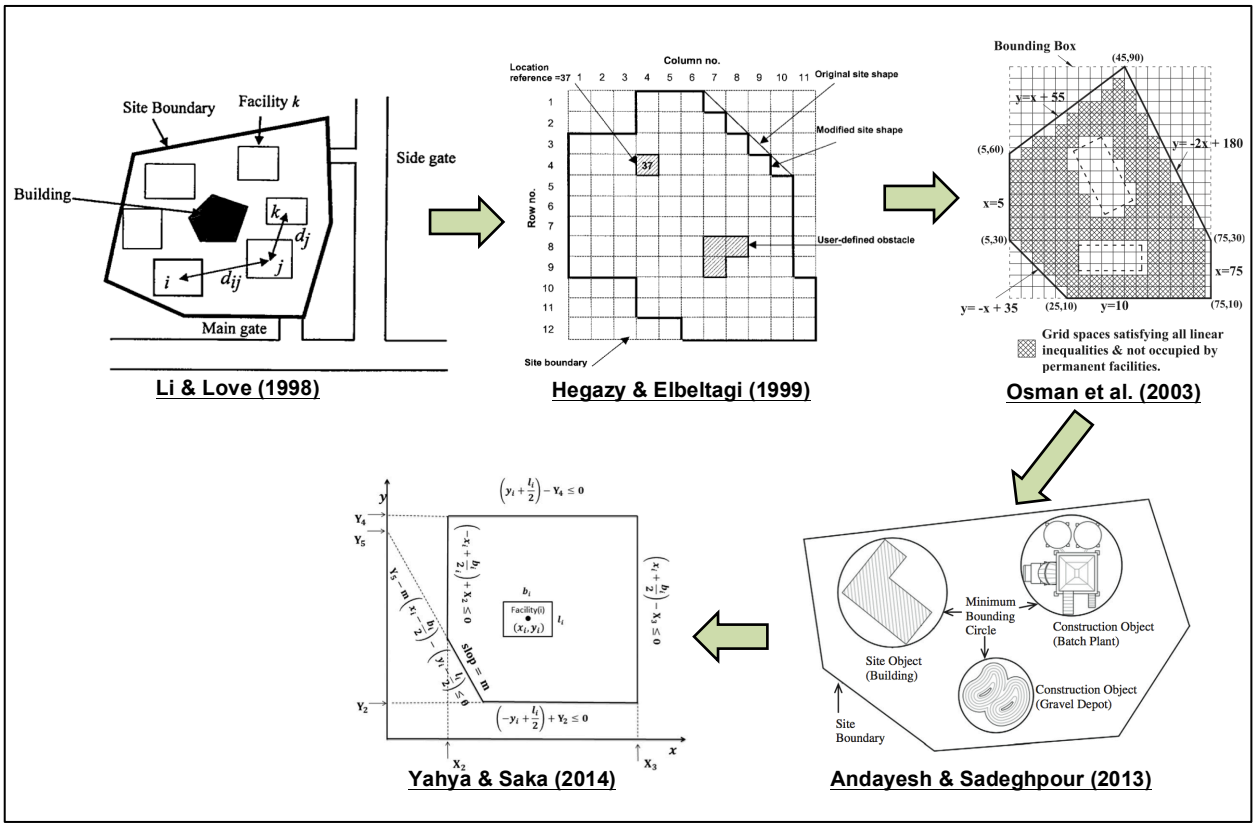

Fig. 1: The development of site and facility representation (modeling)

The use of grids, circles, and rectangles as shown in Figure 1 simplifies the search procedure by decreasing the number of possible choices for the position of objects. However, in reality, the construction site and facilities can acquire any shape and can be located in any place. In strict construction sites, it can be safely claimed that the previously mentioned models face difficulties, that sometimes result in not finding valid results at all, in finding solutions in strict site layouts due to the inefficiency of modeling the facilities through wasted area difference between the "actual" and the "modeled" shapes of the facilities. Moreover, all previous models assumed fixed "static" shapes for the site facilities; while in reality, shapes

of site facilities are “dynamic” in the sense that they can change their geometry to squeeze themselves into the available spaces.

The purpose of this paper is to present a new approach of modeling site facilities to surpass the mentioned shortcomings of previous models and to ensure a more realistic approach to construction site layout problems. The developed model is capable of modeling any regular and irregular site shapes since it utilizes a parametric modeling software (Grasshopper®). The model also has the ability to mimic the “dynamic” behavior of the objects’ shapes through different algorithms. After the site facilities are successfully modeled using the new algorithms presented in this paper, proximity measures and site constraints are added and the model is transformed into an optimization problem that is solved using GA; which is recognized for the development of good results in complex optimization problems with non-linear behavior and large number of variables (Mitchell, 1996).

Grasshopper® is a plug-in for Rhinoceros 3D®, referred to as Rhino®, modeling software. Rhino® is mainly used by architects to facilitate drawing and modeling complex geometrical shapes that are difficult to model using other related software such as AutoCAD 3D® and Revit Architecture®. Grasshopper® enables parametric modeling; allowing users to model and perform analysis on even more complex shapes in Rhino® by inputting parametric data and formulating parametric relationships instead of drawing. Grasshopper® combines the mostly graphical approach of working in Rhino® with the powerful algorithmic techniques found in scripting. Galapagos is an add-in tool in Grasshopper® that employs GA in finding near-optimum solutions within Grasshopper® (graphical algorithm editor).

2 Model Formulation:

A site layout optimization model is mainly formed of three modules: 1) database module, 2) constraints module, and 3) objective function module. The database module is where the user inputs data about the site boundaries, shapes and locations of permanent obstacles, and data about the site facilities –referred to as objects- such as their initial shapes, initial locations and orientations. The constraints module is where the optimization constraints are set. There are three main constraints in the optimization model; 1) collision prevention; where the collision and overlapping between any two facilities or between facilities and obstacles is met with a large penalty added to the objective function. 2) The second constraint is for site boundaries inclusion assurance; where the model ensures that the facilities are bound inside the construction site. 3) The third constraint is the area constraint; where the model ensures that the facilities are between their user-set upper and lower boundaries. This constraint is only present in the “dynamic” objects. The third module is the objective function module; which is for the purpose of measuring the proximities between facilities and calculating the final score, taking the penalties from the constraints into consideration. This module is connected to an optimization engine to find the near-optimum solution.

The used algorithms, especially the ones in the irregular and dynamic shapes, require extensive programming skills to execute on object-orienting programming platforms or even on easier tools such as Microsoft Excel®. For that reason, the modeling capabilities of Grasshopper® were used to execute the algorithms described in this research. Users who wish to execute the discussed algorithms and/or add further developments are preferred to have introductory knowledge of the basic Grasshopper® tools and interface, which is not difficult to obtain.

3 Modeling the site facilities

The site layout facilities acquire different geometrical shapes as seen from a top view. The first step of modeling the site facilities is to model their geometrical shapes. By the utilization of the different tools of Grasshopper®, it is now possible to model complex site facilities. Previous site layout optimization models assumed static shapes for the facilities, meaning that the shapes are the same in all iterations. So if a facility is modeled as a square, it stays a square with the same dimensions during all iterations of the optimization model. In this research, an addition is made by introducing dynamic shapes; where a facility is assumed to acquire a certain shape and in every iteration it changes its shape given certain constraints until it reaches a shape that fits in the unoccupied land on site.

3.1 Types of shapes used for modeling the site facilities (Figure 2):

Construction site facilities presented in this paper are modeled using two different geometrical representation algorithms; static shapes and dynamic shapes as shown in Figure 2. A static shape is where a facility is modeled to acquire a predetermined shape with fixed dimensions. The model supports 6 static shapes: rectangles, triangles, circles, irregular polygons ellipses, and freeforms (polynomial splines, Bezier curves, and NURB curves). A dynamic shape is where a facility is modeled to acquire a shape that changes its dimensions and its form in each run through a series of parametric algorithms. The model introduces two algorithms for modeling two different dynamic shapes; offsetted planar curves and dynamic freeforms.

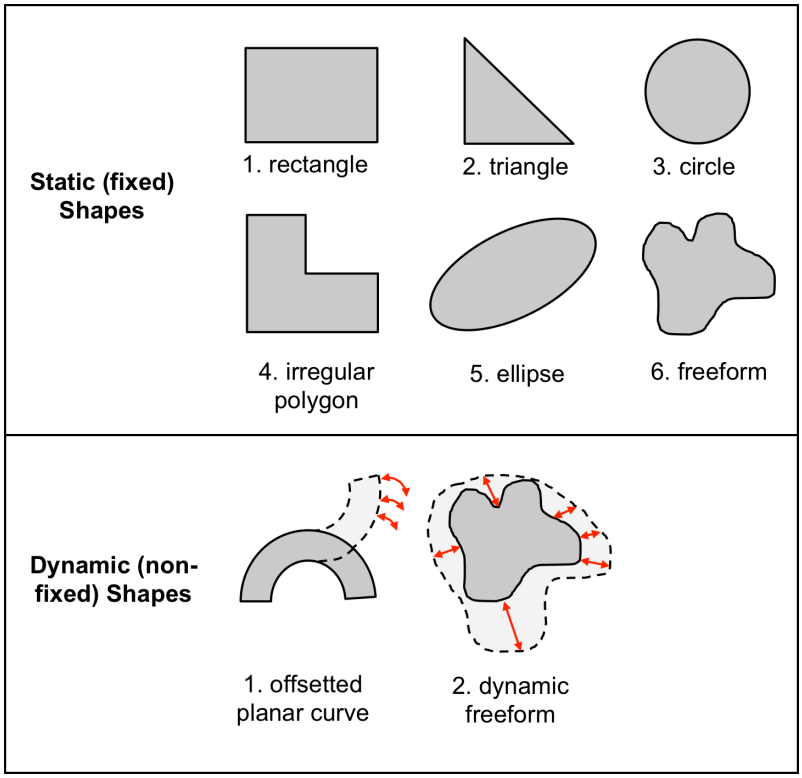

Figure 2: Different shapes for modeling site facilities

The following sub-sections describe each of the static and dynamic shapes and their formulation in Grasshopper®. The formulation of the static shapes for modeling site facilities is very simple on Grasshopper® and does not require the use of many parametric components. However, the formulation of the dynamic shapes requires full understanding of their algorithms, therefore in each of the sub-sections discussing the dynamic shapes, the following order is respected: 1) introduction; covering the use of the shape and the algorithm behind it, 2) steps of modeling; covering how the algorithm is executed on Grasshopper® , 3) variables; stating the variables of the dynamic shape, 4) flexibility; showing samples of the different forms of the dynamic shapes by changing the variables, and 5) verification; providing a verification model to verify the formulated algorithm and ensure the absence of any modeling errors.

3.2 Modeling Static Shapes

A static shape is defined in this research as a geometrical shape with certain parameters; where the parameters do not change in the different iterations, so the initial form and size of the shape is exactly as its final form and size. In other words, a static shape is a geometrical shape that its size-changing and form-changing parameters are not variables in the optimization model. It can be safely acclaimed that all previous research in the site layout modeling assumed static shapes. The main advantage of using Grasshopper® in modeling the facilities' shapes is the parametric inputting method; where any geometry

can be interpreted into parameters; thus creating a very generic pool of shapes that can satisfy all 2D geometries of site facilities.

3.2.1 Modeling Rectangular Objects:

Rectangular surfaces are the mostly used geometrical shapes in site layout modeling. They can be used to model many of the site facilities such as storage areas, caravans, workshops, parking lots, external restrooms …etc. The ***Rectangle*** component is used to create rectangular geometries. The parameters required are 1) P: plane of the rectangle, 2) X; dimension of the rectangle in the X-direction, 3) Y; dimension of the rectangle in the Y-direction, and 4) R; rectangle corner fillet radius.

3.2.2 Modeling Triangular Objects:

Triangular surfaces are not used frequently in site layout modeling. However, it is still beneficial to be able to model triangular facilities as triangular instead of fitting them into rectangular models. The ***Polygon*** is used to create polygons with any number of sides (a triangle is defined as a 3-dided polygon) The required parameters are: 1) P; plane of the polygon, 2) R; radius of polygon (distance from center to tip), 3) S; number of sides of the polygon, and 4) Rf; polygon corner fillet radius.

3.2.3 Modeling Circular Objects:

Circular surfaces are suitable for modeling circular tanks in the site. The ***Circle*** component creates a circle defined by a base plane P and a radius R.

3.2.4 Modeling Irregular Polygons:

An irregular polygon is a polygon with unequal sides and unequal angles. In reality, most of the site layout facilities are irregular polygons, but site layout models model them usually as rectangles; which causes a waste of area and presence of inefficiencies in modeling. To create an irregular polygon

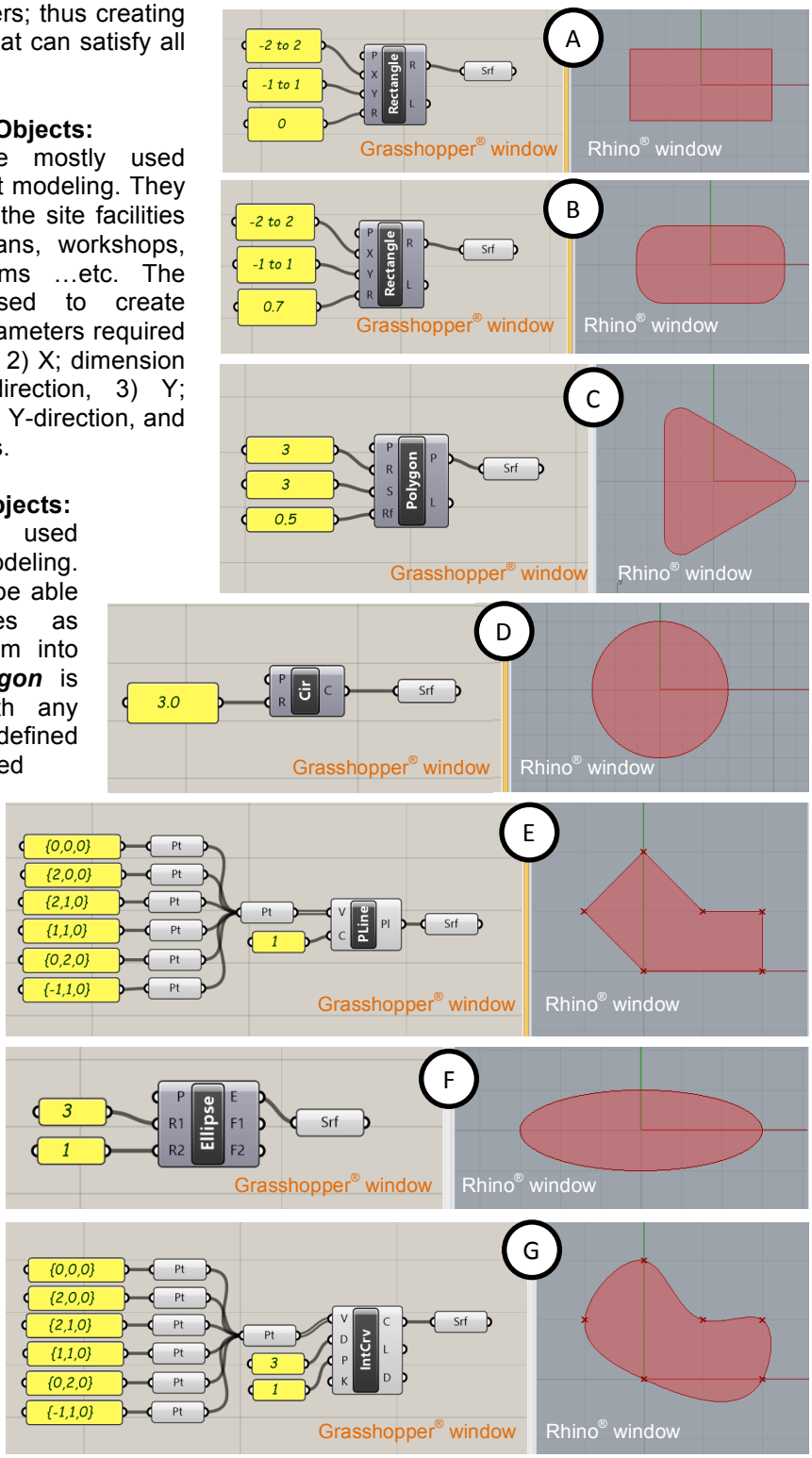

Figure 3: Modeling different geometrical shapes of static site facilities
A) Rectangular object, B) Rectangular object with corner fillet, C) Triangular object with corner fillet, D) Circular Object, E) Irregular polygonal object, F) Elliptical object, G) Freeform object

with any number of sides, the coordinates of its vertices have to be specified first using the ***Point*** parameter. The points are then connected together by a polyline through connecting the ***Point*** parameters to a ***Polyline*** component. It is very important to ensure that the formed polygon is closed by the changing value of the input node marked by C to 1. This algorithm enables the creation of all possible shapes of irregular polygons.

3.2.5 Modeling Elliptical Geometries:

The ***Ellipse*** component in Grasshopper® creates an ellipse defined by a base plane and 2 radii. The ***Ellipse*** component requires 3 inputs: 1) P; plane of the ellipse, 2) R1; radius in X-direction, and 3) R2; radius in Y-direction.

3.2.6 Modeling Freeform Geometries (Static):

Freeforms are the most realistic ways of modeling material piles; especially that materials such as sand and gravels occupy non-linear space. So, it is highly inaccurate to model these material piles as rectangles or linear polygons. A significant addition of this research to the ongoing site layout modeling research is the ability to model one of the most flexible geometrical shapes; which is the freeform. Creating a freeform on Grasshopper® starts be creating the control points. Creating the control points is made through using the ***Point*** parameter for each point, with the corresponding coordinates connected to it. There are several ways of interpolating between points; each type of interpolation results in a different curve. One of the interpolation components is the ***Interpolate*** component; which enables the user to interpolate between the control points at any degree while ensuring that all the points are on the interpolated curve boundary. The main inputs of the Interpolate component are 1) V; the control points, 2) D; degree of interpolation, and 3) P; a Boolean value for whether the curve is close or open (1 for closed and 0 for opened).

3.3 Modeling Dynamic Shapes:

3.3.1 Offsetted Planar Curves (OPC)

The Offsetted Planer Curves OPC algorithm is suitable for modeling caravans in site; since caravans are placed next to each other where their width is fixed. In strict sites with tight available spaces, laying caravans in a straight line is not a luxury, so it is important to provide a tool that not only models the possibility of irregular layout of caravans, but also provide a proposal of the near-optimum form while ensuring the minimum width and a certain range of plan area.

In this algorithm, the purpose is to form a geometrical shape that ensures a certain minimum width while flexible enough to squeeze itself into tight areas and non-uniform boundaries. In such geometrical shape, the width is fixed all over the span. In the OPC algorithm, a main curve – referred to as the ***Spine*** – is formed from several control points; where these points define the shape of the spine using any type of interpolation available in the software such as NURB curves or polyline interpolations. The spine points are constructed relative to each other in order to minimize the number of variables. In other words, each control point is a translation of the preceding control (using the ***Move*** and ***Vector XYZ*** components) point with minimum and maximum X and Y values forming the variables of the algorithm. The spine is then offsetted on both sides forming a width. Then a surface is created between the formed offsets of the spine using the ***Ruled Surface*** component. The formulation of the Offsetted Planar Curves algorithm on Grasshopper is shown in Figure 4.

The variables are the number sliders connected to the movement vectors of the control points (colored in green in Figure 2). By changing the variables, the location and shape of the OPC changes; reflecting very good dynamic flexibility in geometry. The flexibility of this method allows the modeled OPC to fit into complicated zones that are difficult, and almost impossible, to model using traditional modeling tools.

Flexibility: The OPC algorithm produces very flexible surfaces that can take many curved shapes with fixed widths. Figure 5 shows just 5 examples of surfaces made by the OPC algorithm by just changing the number slider values in that are connected the ***Vector*** components of the spine points.

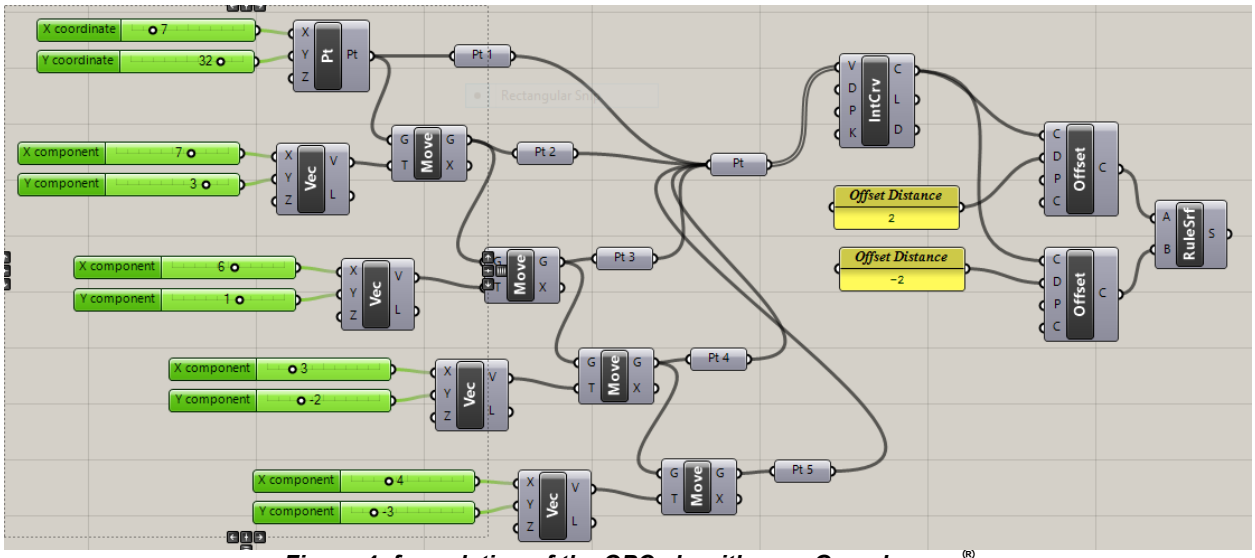

Figure 4: formulation of the OPC algorithm on Grasshopper®

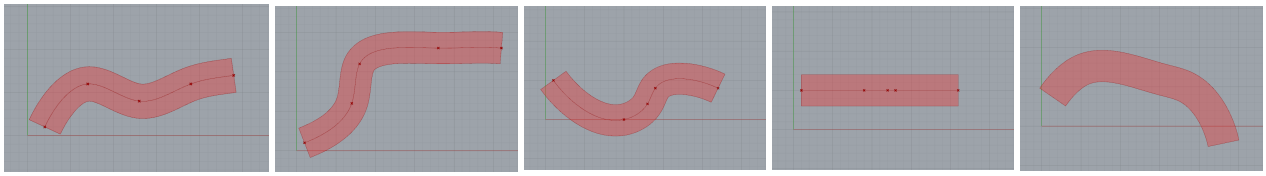

Figure 5: Examples of different possibilities of surfaces modeled using the OPC algorithm

Verification: In order to verify the integrity of the OPC algorithm and ensure the absence of any modeling bugs, a small site layout optimization model was created where the site had four obstacles: A, B, C, and D in Figure 6. The obstacles are non-rectilinear geometrical shapes and spaced relatively close to each other so that it is very difficult to lay the test object using traditional linear modeling methods. The objective function is to minimize the distance between the centroid of the test object and the centroid of obstacle B. The variables were the coordinates of the control points of the spine and the rotation of the OPC. The two constraints were: 1) avoid overlapping between any of the objects on the layout, and 2) keep the offsetted planer surface area between 80 m^2 and 120 m^2. The optimization engine obtained many valid results without errors. Four of the valid results are shown in Figure 6. In each of the solutions, the shape of the object is different due to its adaptive and highly flexible nature. So, it is safe to state that the OPC algorithm can work on a larger scale on real site layout problems.

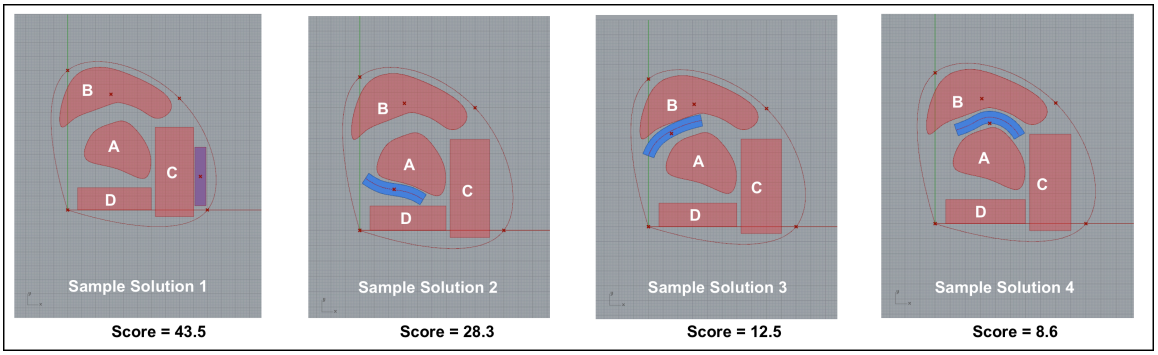

Figure 6: Sample valid solutions for the OPC algorithm verification

3.3.2 Dynamic Freeforms:

A Dynamic freeform is defined in this research as a freeform with flexible control points; meaning that the coordinates of the control points are variables in the optimization model, thus allowing for many flexibilities. In this algorithm, the purpose is to form a geometrical shape that is very flexible– not limited by a specified shape – to squeeze itself into narrow complex areas. Such modeling algorithm is very beneficial for modeling the material piles such as excavation piles and waste piles. Previous site layout optimization models model piles as squares or other linear geometrical shapes. This is not accurate in nature and causes many modeling difficulties such as creating waste of space. In other cases, the models would not find valid solutions in strict site layouts due to the inefficiency of modeling curvilinear shapes and modeling them as approximate linear shapes instead. Not only this algorithm allows the modeling of freeform shapes, it also allows for changes in the shapes in the different model runs to fit into narrow and complex available spaces in the site.

The algorithm works as follows: A point is defined and named the “Reference Point”. Then six surrounding points are defined as “Control Points” of the freeform. The control points are relevant to the reference point. Meaning that the location of each of the control points is defined as the location of the reference point added to a certain vector. The vectors have upper and lower limits for each of the control points. A curve is then interpolated between the control points shaping the freeform. So in order to move the whole freeform, only the reference point needs to be moved and the rest of the control points will automatically move in the same direction since they are all connected to the reference point. The shape of the freeform is changed by changing the different vectors of the control points. So, the variables are the coordinates of the reference point and the vectors of the control points. Figure 8 shows the formulation of the Dynamic Freeform algorithm in Grasshopper®. Figure 7 shows the range of movement of each of the control points relevant to the reference point of the Dynamic Freeform algorithm. According to the current range, there are 48.189x109 possible freeform shapes that can be formed using this algorithm; each run in the model constitutes a different shape.

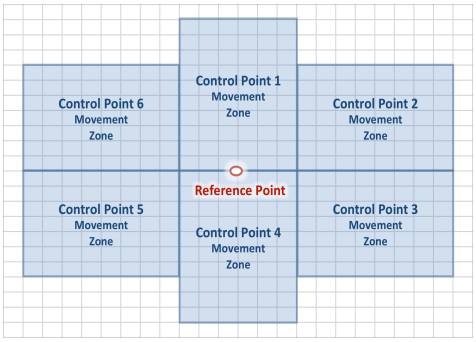

Figure 7: The range of movement of each of the control points relevant to the reference point

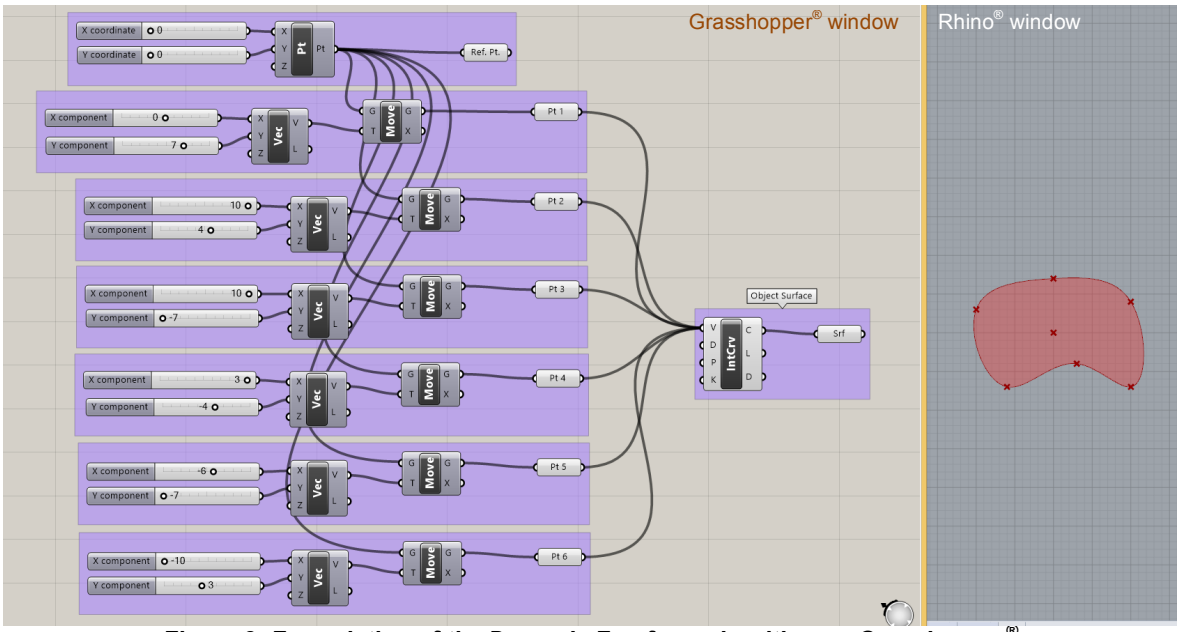

Figure 8: Formulation of the Dynamic Freeform algorithm on Grasshopper®

Flexibility: The dynamic freeform algorithm provides an unprecedented flexibility marking a breakthrough in the site layout modeling research pool. Figure 9 shows a sample of the different formed shapes using the dynamic freeform algorithm. Each run constitutes a different shape. Shapes formed by the dynamic freeform algorithm can fit into complex narrow areas on site.

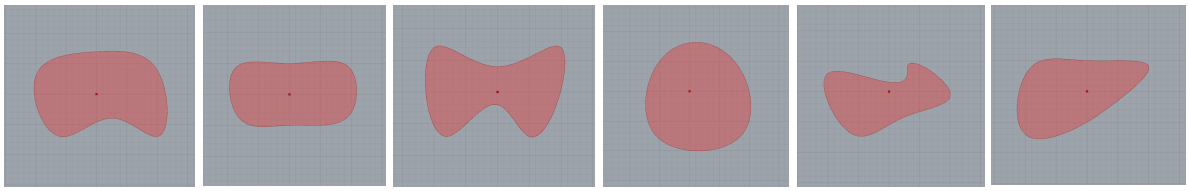

Figure 9: Examples of different possibilities of surfaces modeled using the Dynamic Freeform algorithm

Verification: In order to verify the integrity of the Dynamic Freeform algorithm and ensure the absence of any modeling bugs, a small site layout optimization model was created where the site had four obstacles - colored in rose in Figure 10, formulated that it is very difficult to lay the test object – colored in blue – using traditional linear modeling methods. The objective function is to minimize the distance between the centroid of the test object, which is the dynamic freeform object, and the centroid of obstacle B. The variables were the coordinates of the reference point and the movement vectors, in both X and Y directions, of the control points of the dynamic freeform. The two constraints were: 1) avoid overlapping between any of the objects on the layout, and 2) keep the freeform surface area between 80 m^2 and 120 m^2. It is noticeable from Figure 10 that the test object (dynamic freeform) has a different shape in each of the solutions; allowing for it to fit in complex non-rectilinear spots, which is impossible to model using previous modeling algorithms.

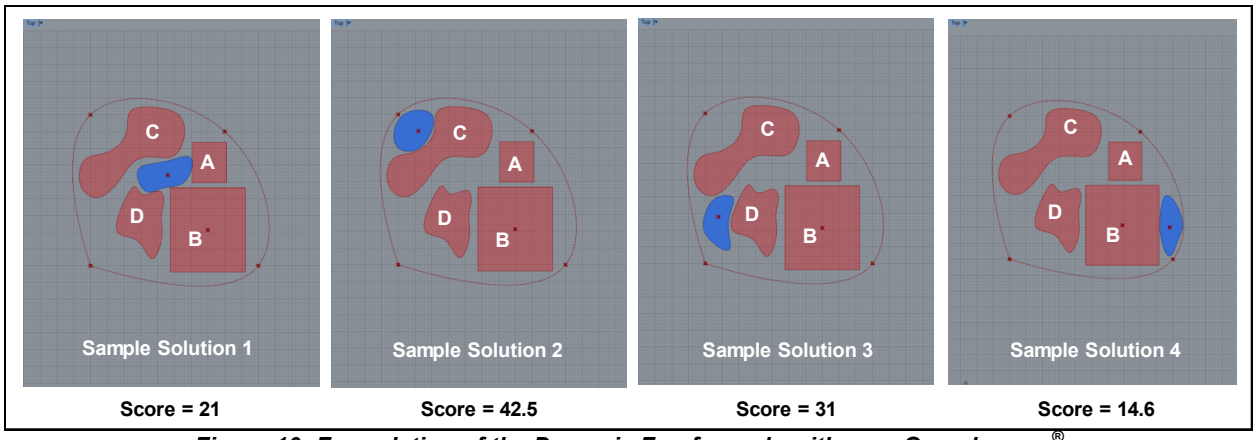

Figure 10: Formulation of the Dynamic Freeform algorithm on Grasshopper®

3.4 Collision Prevention Constraint Using the Area Union Approach

The algorithm used for preventing collision between site facilities is developed in Grasshopper® using an area union approach instead of the already built-in collision detection component because it fails to detect inclusion (full overlapping, where a facility is fully enclosed in another facility). The area union approach is based on a simple, yet novel idea, which is very simple in formulation as well. The algorithm takes advantage of the ***Region Union*** component of Grasshopper® which unions several surfaces (objects) together. The algorithm follows the following steps to detect collision:

1. Specify the area of each object [A_i].
2. Get the summation of the areas of the objects [ΣA_i].
3. Union the surfaces of the objects into one surface.
4. Specify the area of the resultant of the union of surfaces [A_u].
5. There are only two possible outcomes:
    a. $\Sigma A_i = A_u$ → means that there is NO collision → Do not add penalty
    b. $\Sigma A_i > A_u$ → means that there is collision → Add penalty to the objective function

4 Conclusion:

An accurate representation of the construction site is important for site layout modeling, as it enables the development of more realistic and efficient layouts. Previous research efforts used grids, circles, and rectangles to model site facilities; which simplifies the search procedure. However, in reality, the construction site and facilities acquire more complex and flexible shapes. This paper presented a new and more realistic approach of modeling site facilities. It presented the algorithms of modeling different "static" geometries such as rectangles, triangles, circles, irregular polygons, ellipses, and freeforms, and "dynamic" geometries such as the offsetted planar curves and the dynamic freeforms. Dynamic geometries are where a facility is assumed to acquire a certain shape and in every iteration, it changes its shape given certain constraints until it reaches a shape that fits in the unoccupied land on site. The model utilized a parametric modeling software (Grasshopper®) for modeling. Proximity measures and site constraints are added and the model is transformed into an optimization problem that is solved using GA to obtain near-optimum locations and shapes of site facilities. The paper also introduced a new approach for collision prevention using an Area Union algorithm. More developments shall be taking place in the model such as introducing new distance measurement techniques, control points flexibilities, new dynamic geometrical algorithms, buffer zones as soft constraints, and selective zoning of facilities to minimize the number of variables in the model.

References:

[1] Andayesh, M., Sadeghpour , F. (2013). "Dynamic site layout planning through minimization of total potential energy", Automation in Construction, 31, 92-102
[2] Cheng, M.Y. and O'Connor, J.T. (1996). "ArcSite: Enhanced GIS for construction site layout", Journal of Construction Engineering & Management, ASCE, 122(4), 329-336
[3] Elbeltagi, E. and Hegazy, T. (2001) "A hybrid AI-Based system for site layout planning in construction", Computer-Aided Civil and Infrastructure Engineering, Blackwell Publishers ,16(2), 79-93
[4] El-Rayes, K., Said, H. (2009). "Dynamic Site Layout Planning Using Approximate Dynamic Programming", Journal of Computing in Civil Engineering, 23 (2), 119-127
[5] Hegazy, T. and Elbeltagi, E. (1999). "EvoSite: Evolution based model for site layout planning", Journal of Computing in Civil Engineering, ASCE, 13(3), 198-206
[6] Heng Li and Love, P.E. (1998). "Site level facilities using genetic algorithms", Journal of Computing in Civil Engineering, ASCE, 12(4), 227-231
[7] Khalfallah, A., El-Rayes, K. (2002). "Automated multi-objective optimization system for airport site layouts", Automation in Construction, 20, 313-320
[8] Mitchell, M. (1996). An introduction to genetic algorithms. Cambridge, Mass.: MIT Press.
[9] Ning, X., Lam, K., Lam, M. (2011). "A decision-making system for construction site layout planning", Automation in Construction, 20, 459-473
[10] Osman, H., Georgy, M., Ibrahim, M. (2003). "A hybrid CAD-based construction site layout planning system using genetic algorithms", Automation in Construction, 12, 749-764
[11] Tommelien, I.D., Levit, R.E., and Hayes-Roth, B. (1992). "SitePlan model for site layout", Journal of Construction Engineering & Management, ASCE, 118(4), 749-766
[12] Xu, J., Li, Z. (2012). "Multi-Objective Dynamic Construction Site Layout Planning in Fuzzy Random Environment", Automation in Construction, 27, 155-169
[13] Yahya, M., Saka, M. (2014). "Construction site layout planning using multi-objective artificial bee colony algorithm with Levy flights", Automation in Construction, 38, 14-29
[14] Yeh, I-Cheng. (1995). "Construction-site layout using annealed neural network", Journal of Computing in Civil Engineering, ASCE, 9(3), 201-208
[15] Zouein, P.P., Harmanani, H. and Hajar, A. (2002)."A genetic algorithm for solving the site layout problem with unequal size and constrained facilities", Journal of Computing in Civil Engineering, ASCE, 16(2), 143-151