INTEGRATING KNOWLEDGE-BASED SYSTEM WITH GEOMETRIC DATA FOR SITE LAYOUT PLANNING

Mohamad Jamil Sulaiman

National CAD/CAM Centre
SIRIM Berhad
1, Persiaran Dato' Menteri, Section 2, P.O. Box 7035
40911 Shah Alam, Selangor Darul Ehsan
Malaysia

RINGKASAN: *Susun atur untuk sesuatu tapak merupakan satu proses yang kompleks. Ianya memerlukan analisa maklumat yang terperinci daripada pelbagai disiplin (seperti rekabentuk, perancangan pembinaan, penganggaran dan sebagainya). Ini melibatkan pengenalpastian, penempatan dan penyusunan kemudahan sementara, bangunan dan bahan-bahan, mengikut kesesuaian yang diperlukan dalam melaksanakan proses pembinaan pada setiap peringkat pembinaan. Kertas kerja ini akan menerangkan cadangan algoritma yang digunakan dalam membangunkan Intensite (Intelligent Site Layout Planning) yang dihasilkan daripada perlaksanaan sistem yang merangkumi tindakbalas yang sepatutnya, penilaian dan pengalaman yang berasaskan kepakaran yang sedia ada. Maklumat geometri daripada CAD (Rekabentuk Berbantukan Komputer) secara automatik akan dipetakan secara talian kepada suatu sistem yang berasaskan pengetahuan untuk mengoptimumkan perancangan susun atur untuk sesuatu tapak.*

ABSTRACT: The site layout is a complex process, requiring the analysis of large amounts of information from several disciplines (such as design, construction planning, estimating and others). It involves identifying, allocating and positioning temporary facilities (TFs), plant and materials, to best suit the needs of the construction process at each stage of construction. This paper describes the proposed algorithms used in the development of *InteSite* (Intelligent Site Layout Planning) that has resulted in the implementation of a system that embodies the intuitive responses, judgements and experience of experts in its knowledge-base. Geometric information from CAD system is automatically mapped into knowledge-base system at real time for optimising site layout planning.

KEYWORDS: Knowledge-based system, temporary facilities, virtual lines.

INTRODUCTION

The construction industry has been criticised for being slow to accept and apply modern management methods, for the planning and execution of projects, and this is said to explain, in part the poor construction performance (Alshawi and Putra 1994; Oglesby *et al.,* 1989). In the field of construction site planning, the problems of site layout have not been fully addressed, and can be considered as one of the most neglected aspects of the construction industry. The typical attitude is "This will be done as the project progresses", concealing the fact that - as most contractors and construction managers admit - they are too busy to plan ahead (Handa and Lang, 1988). The construction industry, by comparison, has found little use for spatial analysis, a probable factor being the perceived complexity of the construction environment, involving, as it does, the movement of plant and labour and such constraints as climatic conditions, site geography, existing services and regulations (Tommelein and Zovein, 1993; Tommelein, 1994). Space considerations, however, do directly influence construction costs, which depend on choice of construction methods, on plant selection, and on the quality of construction planning and material management. The whole project process requires extensive information, and the professional teams at each stage are faced with difficulties in collecting the information (Alshawi and Sulaiman, 1995). So further research attention to site layout planning, which is essentially the good management of space, is well justified in terms of its potential for cost saving and for improving intercommunication between the various practitioner teams involved in a complex project.

OBJECT-ORIENTED KNOWLEDGE-BASE

KAPPA-PC is an object-oriented knowledge-based (OOKB) system shell that supports objects, hierarchies, methods, interactive graphical interfaces, dynamic data exchange and others. It uses classes and instances to represent objects. Objects' behaviours are represented in methods, functions and rules. Methods and functions are used to specify the behaviour of slots in each class or instance. Rules are also provided to deal with knowledge framed in terms of 'IF-Then' decisioning. The followings are some of the KAPPA-PC features:

a) Interface-supported by means of menus, images and which have made development of the interface a straightforward task.

b) Developer environment-has its own developed environment for which developers have built applications using the in-built tools (i.e. various editors and the KAL interpreter). KAL is a KAPPA-PC's internal language that proved adequate and convenient for managing and querying the knowledge-base.

c) Ability to integrate-integrates/interacts with other software through DDE (Dynamic Data Exchange). DDE enables the exchange of data between different applications.

Transferring Geometric Information To Knowledge-base

Bare geometric information (lines) created in a wire-frame CAD environment, must be extracted, interpreted and exported to the site layout module. For example, a user-drawn site boundary is described in AutoCAD by a list of entities in the following internal format.

[(-1. <Entity name: 6000001a>) (0 . "LINE") (5 . "A6B")(8 . "LAYER1") (10 563.000000 484.000000) (11 1162.000000 745.000000) (210 0.0 0.0 0.0)]

Where (-1) is the group code for the entity name, (0) identifies the entity type, and (5), the entity handle. An 'entity handle' is the entity's unique ID. This is an important data element on which the system relies to control updates at the central core (when the element is graphically updated or deleted). Group code (8) refers to layer (10) to the start-point co-ordinates, and (11) to the end-point co-ordinates. (210) Codes the line's orientation. A function - the Object Interpreter - captures the graphically entered elements and transfers them to the site layout module, the first step being to capture the entity's handle and co-ordinates. All the information is stored in the respective class and attributes in the object structure.

CHECKING SPACE FOR FACILITIES

Determining Space on Site

InteSite automatically defines the available space (in terms of how much and where) from the CAD drawing and places the required (TFs; temporary facilities) at their optimal locations. To do so it is necessary to first consider spatial ('far-from', 'close-to' and others) relationships.

When a user draws a building parameter in AutoCAD, each line is assigned with a unique ID (such as 'BLD_4E11'). This in turns instantiates the 'Building_L' class in the site layout module. Each of these lines is then evaluated with respect to the site boundary for two reasons:

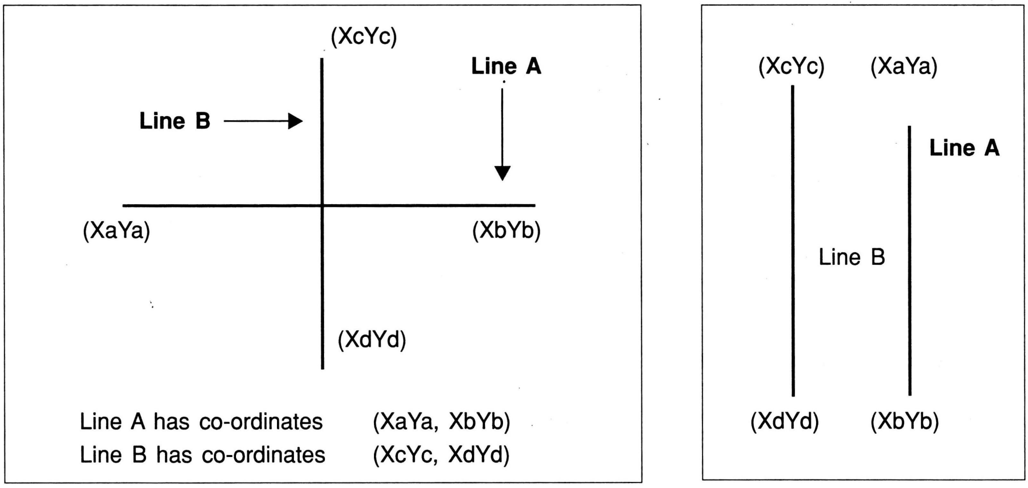
a) To find the maximum distance achievable between the building location and the site boundary. This is required for locating objects that must be 'far-from' the building (work place).

b) To calculate the space available for locating the facilities.

Spaces are defined by projecting 'virtual lines' from all building parameters. Two from each corner and one from the middle. These line are used to calculate the distance from their original projection points to the intersection of a new line. This distance is the key determinant of available spaces on site, and is used for calculating distances between TFs and the site boundary (and the access road), and for checking the locations of plant.

Figure 1a illustrates an intersection between 'line A' and 'line B', in the case shown in Figure 1b the lines do not intersect. An algorithm has been developed for determining whether 'intersect?' is True or False. 'Intersect?' is when Line A satisfies the following conditions;

$(Ya < Yc \text{ and } Yb > Yd) \text{ and } (Xa < Xc \text{ and } Xc > Xd)$ .OR.
$(Ya > Yc \text{ and } Yb < Yd) \text{ and } (Xa > Xc \text{ and } Xc < Xd)$ .OR.
$(Ya < Yc \text{ and } Yb > Yd) \text{ and } (Xa < Xc \text{ and } Xc < Xd)$ .OR.
$(Ya > Yc \text{ and } Yb < Yd) \text{ and } (Xa > Xc \text{ and } Xc > Xd)$


(XcYc)
Line A
Line B
(XaYa)
(XbYb)
(XdYd)
Line A has co-ordinates (XaYa, XbYb)
Line B has co-ordinates (XcYc, XdYd)
(XcYc)
(XaYa)
Line A
Line B
(XdYd)
(XbYb)

(a) Case One: 'Intersect'

(b) Case Two: 'No Intersect'

Figure 1. *Theory of Intersection*

Calculating distance between TF and Site Boundary and Access Road

Some objects have to meet two conditions (a waste skip is ideally 'close-to' the workplace and to the access road). To achieve the optima it is necessary to calculate the distances between objects that must be 'close-to' the site boundary, AND 'close-to' the access road, which is done as follows:

a) A virtual line, one from each side (V1, V2, V3 and V4) is projected from the centre of the TF until it reaches the maximum length (Figure 2).

b) When an intersection occurs, then the distance can be calculated as follows:
Distance = Length of the intersected virtual line - $^{1}/_{2}$ (Length or Width of the object)

Where; $^{1}/_{2}$ Length is when the angle of virtual line is at 0 or 180 degree.

$^{1}/_{2}$ Width is when the angle of virtual line is at 90 or 270 degree.

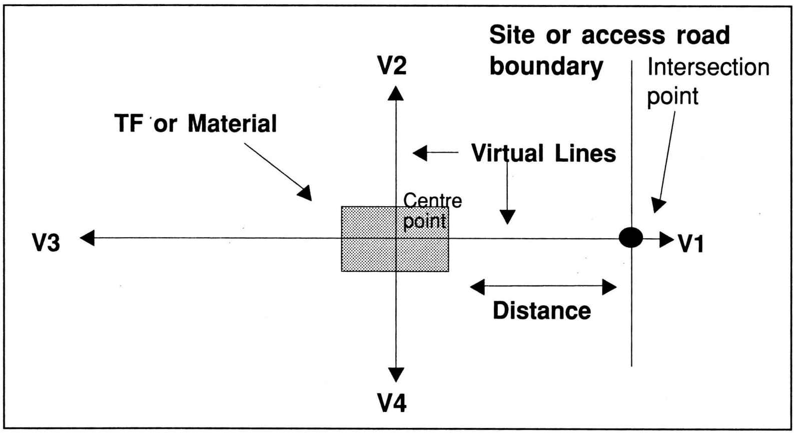


Figure 2. *Virtual line from TF or Material*

Relating plant to an access road

Something moveable, (i.e. lorries), must be located on the access road. The procedure for resolving these cases begins by projecting a virtual line with a maximum predetermined length (=longest side of site boundary) on each side of the plant object. Figure 3 shows how a lorry is considered to be 'inside the access road' when all the four virtual lines (V1,V2,V3,V4) intersect with the access road lines (P1,P2,P3,P4), ELSE it is considered to be 'outside the access road'. Figure 4a shows that the lorry is now 'outside the access road' - only two intersection points have been determined.

If there are four intersection points (two on each virtual line - as shown Figure 4b), then the object (lorry) is also considered to be 'outside the access road'. The algorithm considers only the closest intersection point on each virtual line (P1 and P4).

Detecting Clashes between Facilities

The graphical representation of TF, plant and material are used to check the clashes between them. It is important to detect clashes to (i) avoid clashes between objects automatically placed by and (ii) to alert users of potential clashes when they manually relocate object around the site. The algorithm for clash detection implemented in the main core is as follows:

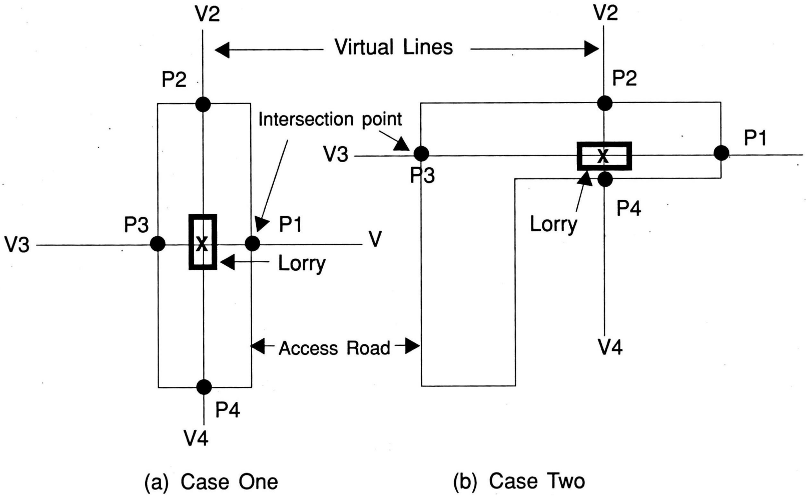
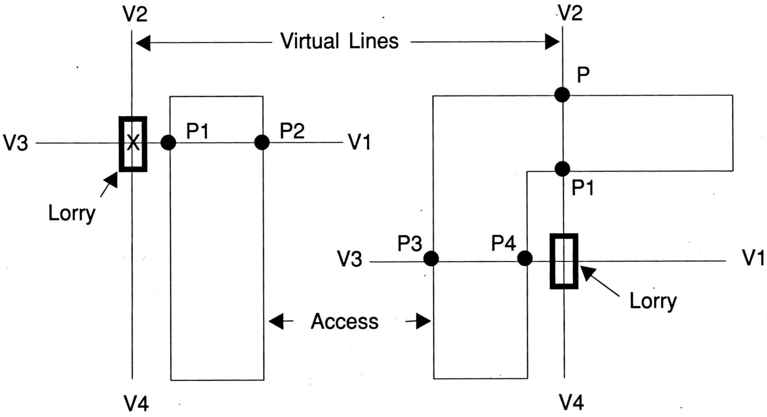


Figure 3. *Lorry within an access road*



(a) Case One:
P1 is considered as the intersection point

(b) Case Two
P1 & P4 are considered as the intersection point

Figure 4. *Lorry outside an access road*

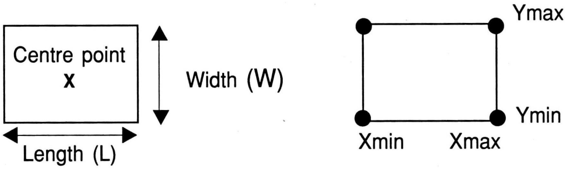


Figure 5. *Property of Graphical Object*

From the dimension of objects (Figure 5), a maximum of X and Y, and the minimum of X and Y values can be calculated as follows:

Xmin = CP - L/2　　　　Ymin = CP - W/2
Xmax = CP + L/2　　　　Ymax = CP + W/2

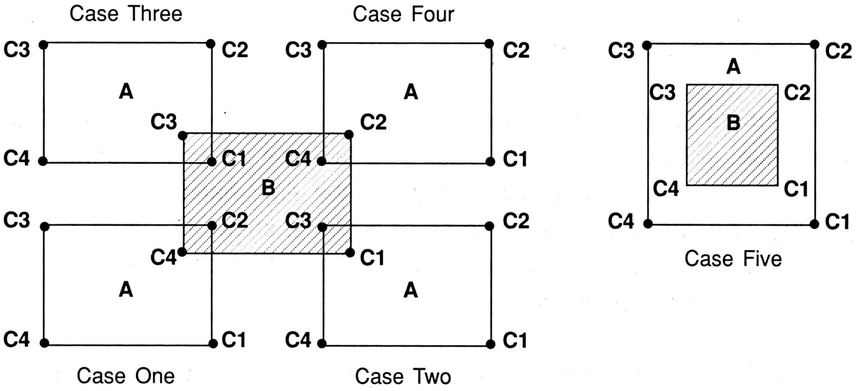


Figure 6. *Clash Detection*

The developed algorithm can handle the five possible types of clashes (Figure 6).

Case One

When an object A (corner C2) is positioned at the bottom left corner (C4) of object B.

X Amin ⩾ X Bmin and X Amin ⩽ XBmax and Y Amax ⩾ YBmin and YAmax ⩽ Ybmax

Case Two

When an object A (corner C3) is positioned at the bottom right corner (C1) of object B.

X Amin ⩾ X Bmin and X Amin ⩽ XBmax and Y Amix ⩾ YBmin and YAmin ⩽ Ybmax

Case Three

When an object A (corner C1) is positioned at the top left corner (C3) of object B.
X Amax ⩾ X Bmin and X Amax ⩽ XBmax and Y Amin ⩾ YBmin and YAmin ⩽ Ybmax

Case Four

When an object A (corner C4) is positioned at the top right corner (C2) of object B.
X Amax $\geqslant$ X Bmin and X Amax $\leqslant$ XBmax and Y Amax $\geqslant$ YBmin and YAmax $\leqslant$ Ybmax

Case Five

When an object A is positioned at *or about* the same location as object B.
X Amax $\geqslant$ X Bmin and X Amax $\leqslant$ XBmax and Y Amin $\geqslant$ YBmin and YAmin $\leqslant$ Ybmax

Assigning Co-ordinates to Facilities

When one or more facility objects have been selected, they must be placed at the best possible location. In Figure 7 Object 1 (Guard house) has a higher priority than Object 2 (Store), but both are required to be located on the right hand side of the main gate ('SITE_4E30'). How these objects are to be arranged is shown in Figure 7. An algorithm has been developed to position the objects at the optimum position (relative to site boundary, access road and building parameter);

1. Find the attributes of the boundary line, for example, the co-ordinates of (x1,y1), (x2,y2) and an angle β of that line (SITE_4E30).
2. Calculate the distances from the centre of the objects i.e. a1, a2, b1 and b2.

The length for a1, a2 b1 and b2 are calculated as follows (Figure 7);
H1 = Square Root of $(0.5W1)^2 + (0.5L1)^2$
$\sigma 1 = \tan^{-1} W1/L1$
$\alpha 1 = \sigma 1 + \beta$
b1 = sin α1 (H1)
a1 = cos α1 (H1)

H2 = Square Root of $(0.5W2)^2 + (L1)^2 + (K)^2 + (0.5L2)^2$

$$\sigma 2 = \tan^{-1} \frac{0.5W2}{L1 + K + 0.5L2}$$

K= Spacing between the two objects. For this study, the spacing between the two objects is taken to be 1m.

$\alpha 2 = \sigma 2 + \beta$
b2 = sin α2 (H2)
a2 = cos α2 (H2)

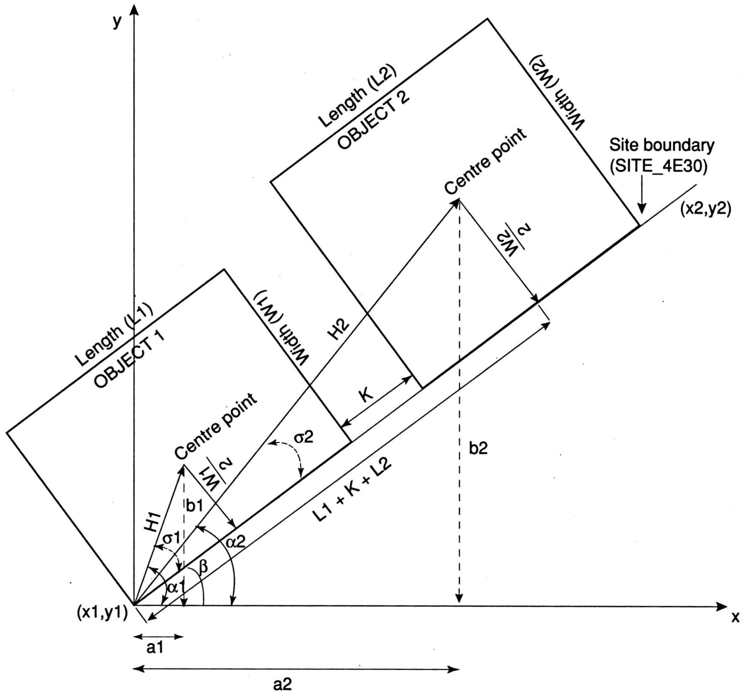


Figure 7. *Positioning two objects at an object 'Site Boundary Line'*

Therefore the positions of object1 and object2 are calculated as follows:
The co-ordinate of object 1 is:
$x = x1 + a1$
$y = y1 + b1$

The co-ordinate of object 2 is:
$x = x1 + a2$
$y = y1 + b2$

SUMMARY

This paper describes how the developed prototype of InteSite was implemented and its interaction with users. It features an object-oriented development to map the developed object-oriented models. KAPPA-PC has been used as the OOKB environment with the geometry information input from AutoCAD. The data are analysed in OOKB accordance to domain expert and the output can then be viewed either in 2D (in AutoCAD) or 3D (in virtual reality). The proposed algorithms for optimising site planning has been proven workable, this new approach promises to bring closer the provision of practically useful site planning system.

REFERENCES

Alshawi, M. and Putra, C.W.F. (1994), Conceptual modelling for 3D simulation of design and construction. *Third Regional Conference on Computing Application in Civil Engineering,* Kuala Lumpur, 2-4 August, 1994.

Alshawi, M. and Sulaiman, M.J. (1995), Applying structured process analysis to site layout planning, *Proceedings of the Sixth International Conference on Computing in Civil and Building Engineering,* Berlin, 1995, pp 477-483.

Handa, V. and Lang, B. (1988), Construction site planning, *Construction Canada,* **85** (5): pp 43-49.

Oglesby, C., Parker, H. and Howell, G. (1989), Productivity improvement in construction. McGraw-Hill-Book Company, New York, 1989,

Tommelein, I.D., and Zovein, P.P. (1993), Interactive dynamic layout planning, *Journal of Construction Engineering and Management*, ASCE, **119** (2): pp 266-287.

Tommelein, I.D. (1994), Material Handling and site layout control, *Automation and Robotics in Construction X1,* Proceedings of the 11th International Symposium on Automation and Robotics in Construction (ISARC), Brighton, UK, pp 297-304.