Redefinition of geometrical components to specify kinematically undetermined behaviour

H. Hofmeyer
Eindhoven University of Technology, Department of Architecture, Building, and Planning, The Netherlands

F. Gelbal
Eindhoven University of Technology, Department of Mathematics and Computer Science, The Netherlands

ABSTRACT: A method for the redefinition of geometrical design components, to be used for specifying kinematically undetermined behaviour, is presented. It starts with checking all line-line combinations for intersection or for a line-line combination being collinear. If an intersection has been found, it is used to generate additional lines such that both old lines are split up at their intersection, resulting in new lines. For lines being collinear a similar approach is followed. Hereafter, all line-area combinations are checked. If certain conditions are met, new intersection points and lines are generated in the area to provide new areas that split up the old area correctly. The procedures (line-line and line-area) are repeated until convergence. Finally, pattern recognition is used to find all areas from the intersection points within an original area. A C++ program and a number of examples verify the method and test its efficiency.

1 INTRODUCTION

1.1 *Problem definition*

Recently, the idea of a research engine has been developed, in which spatial building designs are modified or transformed into structural building designs and vice versa by means of a cyclic procedure (Hofmeyer 2007), Figure 1. The research engine provides support in two domains. First of all, the transformation or modification methods ("Trans. selection" in Figure 1) can be varied and the resulting spatial and structural design evolution can be followed (by means of the "Measure" in Figure 1), resulting in a study of the design process. Secondly, the actual spatial and structural designs provide the designer with possibly new solutions, resulting in a study of generative design. The concept of the research engine is general and can be interpreted in several ways, e.g. as a conventional architectural or structural design technique, as structural optimization, or as multi-disciplinary optimization (Hofmeyer & Kerstens 2008).

Within the research engine, the transformation from a spatial to a structural design consists of four sub transformations: (1) from spatial design to structural topology, (2) from structural topology to mechanical model, (3) from mechanical model to finite element model, and (4) from finite element model to design recommendations (Hofmeyer & Bakker 2008). The first sub-transformation can be a set of spatial-structural transformation rules that adds geometrical components (lines and areas, which later will become structural elements like columns, beams, and plates) to a spatial design. Another sub-transformation uses a finite element model to specify the kinematically undetermined behaviour of the geometrical components by finding the null-space of the system's stiffness matrix (Hofmeyer & Russell 2009), a solution to a problem also signalled in Rotke 1998.

The problem under investigation here is that a finite element model can only be initiated if the geometrical components are redefined correctly for component connections and intersections, as shown in Figure 2.

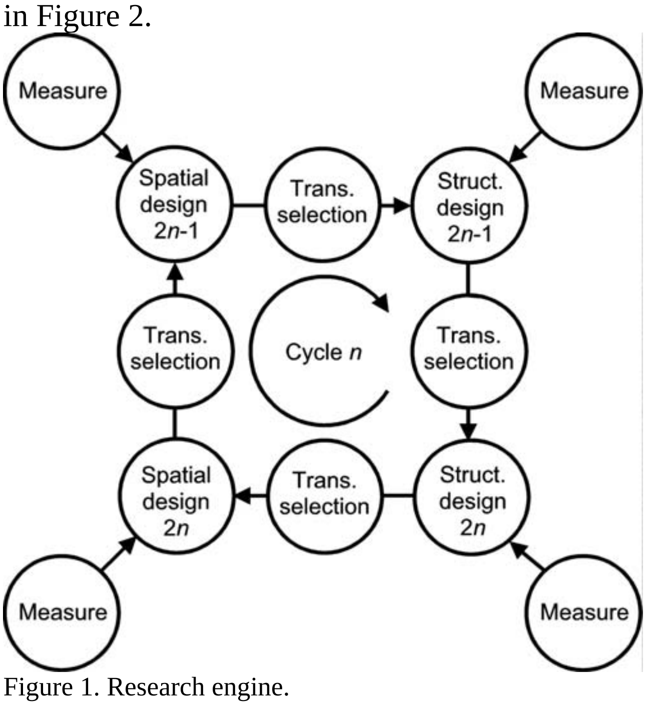

Figure 1. Research engine.

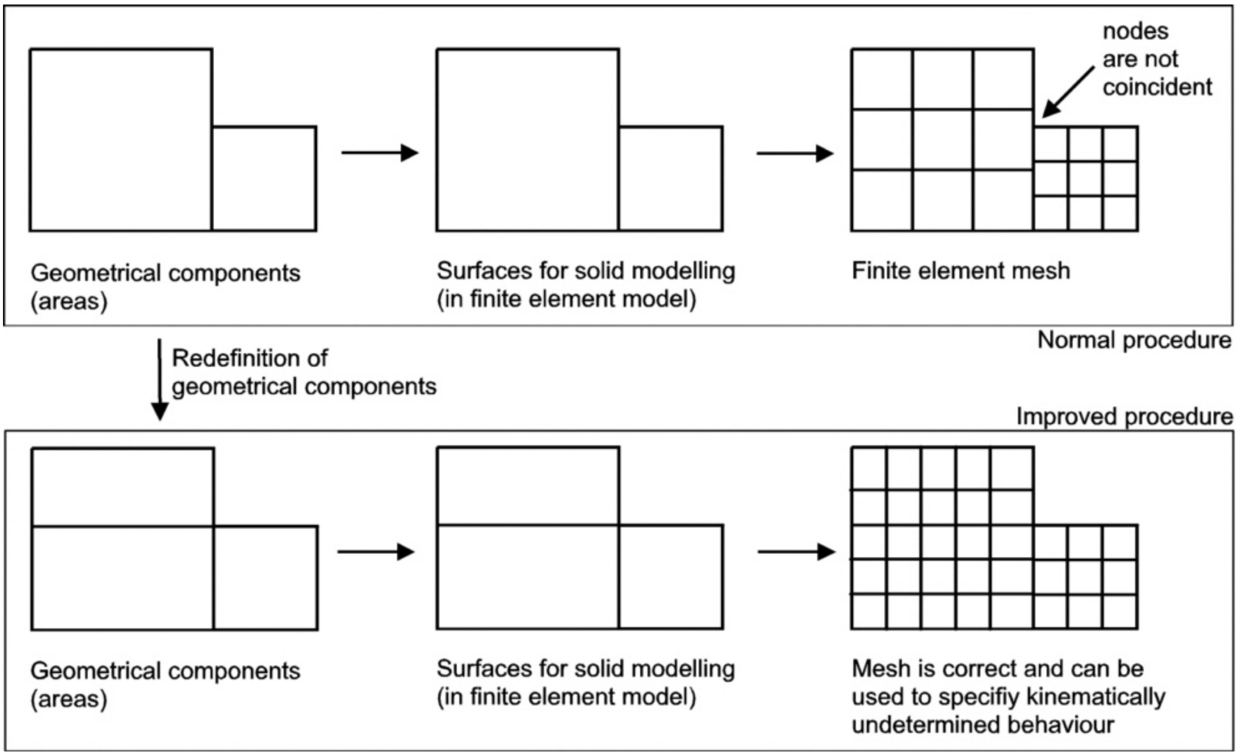

Figure 2. Redefinition of geometrical components.

In this paper a method for this redefinition (three-dimensional, but only for orthogonal oriented lines and areas) will be presented. As will be shown by the application of a C++ program to a number of examples, the method presented functions correctly. However, the problem can also be solved by a slightly different method (Van Roosmalen 2009). In future, these two methods could be compared for correctness and efficiency followed by selecting the optimal method for use within the research engine. Hereafter, for further development of the research engine, a method should be developed that generates a kinematically determined structural design that is representative for building designs as found in practice, see also (Hofmeyer & Russell 2009).

1.2 *Existing research: framework and algorithms*

Considering the research engine, in the field of AEC-field (Architecture, Engineering, and Construction) many research projects have been carried out to investigate the multi-disciplinary character of the field and to develop computer aided tools to support the design processes involved (e.g. Fenvens et al. 1994). In this paper, within the multi-disciplinary design process, only the disciplines of spatial design and structural design are part of the problem definition. Related research, thus on the disciplines of spatial design and structural design, can be divided in three groups. The first group is descriptive research that develops data models, which formalize data and their relationships regarding specific aspects of the design process. Related to this paper are data models that have been specifically developed to relate spatial and structural design (Khemlani et al. 1998, Matthews et al. 1998, Eastman & Jeng 1999, Rivard & Fenves 2000, Scherer & Ghere 2000, Mora et al. 2006). The second group is generative research that yields programs, procedures, or concepts for generating spatial and/or structural design solutions. The oldest but still active field in this group is that of space-allocation that transforms building requirements into a spatial design (e.g. Kotspoulos 2006). For structural design, a distinction should be made between research that optimises an existing structural design by means of expert systems, form-finding or optimization (e.g. Rafiq et al. 2003, Bletzinger & Ramm 2001, Kocaturk et al. 2003) and research that results in the actual one-way transformation and evaluation from spatial to structural design (Rafiq & MacLeod 1998, Fenves et al. 2000). For most research in these two groups, the basic underlying idea is that in the design process a more or less one-way path runs from spatial to structural design. However, the building design process can also be modelled with a more cyclic approach, as shown in Figure 1. A start is made in cycle n by the transformation of spatial design $2n$-1 into structural design $2n$-1, which is often carried out by a structural engineer. The resulting structural design $2n$-1 will be subject to improvement, for example by expert views of other structural engineers or by optimization techniques. This optimised structural design $2n$ will be given to the architect and he will then adjust the spatial design $2n$-1 to fit the structural design, which gives spatial design $2n$, or to fulfil other requirements from the building plan yielding spatial design 2*n*-1 for the next cycle (*n* increases by 1). The resulting design cycle -as shown in Figure 1- is defined as "interaction between spatial and structural design" and the use of this model of the design process in the research engine is justified by many research projects in the third group, namely on the support of multi-disciplinary design processes, e.g. a building design project can be seen as a sequence of views and dependencies from several disciplines (Haymaker et al. 2004).

Specific concerning the problem as defined in this paper, grid and/or mesh-generation is a relatively new published research domain, initiated by the work of Thompson, Warsi, and Mastin (Thompson et al. 1985) stated as such by Frey & George (2000), who present the domain with an more applied approach. Knupp and Steinberg present the domain using a more fundamental approach (Knupp and Steinberg 1993). Within this research domain, the problem as defined in section 1.1 and figure 2 can be regarded as a multiblock problem: A geometry is complex and is partitioned in partitions where the local meshing process applies (Frey & George 2000). Following Frey & George 2000, two kinds of partitions may be considered. The first is conformal in itself (the bottom row of geometries in figure 2) whereas the other does not require such a property. Normally, the partition process is carried out manually, but computer assisted for consistency issues like numbering, line division etc, thus defined as a semi-automatic method. Also, the use of boundary elements or multi-point constraints can provide a workaround (Aus et al. 1996). To automate the partitioning of a geometry such that it is conformal "medial axis tranform" can be used, although this is more complex for 3D than for 2D situations. Given this information, the problem in this paper can be defined as a non-conformal partitioned geometry that should be transformed in a conformal partitioned geometry. The geometry consists of lines and quadrilateral areas only and all elements are orthogonal oriented relatively to each other. The authors could not find literature that treats this specific problem and thus developed a method for it as outlined below.

1.3 *Method overview and paper contents*

The method presented in this paper starts with checking all line-line combinations for intersection or for a line-line combination being collinear. If an intersection has been found, this intersection is used to generate additional lines such that both old lines are split at their intersection, resulting in new lines. For lines being collinear a similar approach is followed. The foregoing procedures are presented in section 2. Hereafter, as presented in section 3, all line-area combinations are checked for three conditions: (1) a line is intersecting a boundary line of an area; (2) one of the line end-points is within the area; (3) a line intersects the area and is normal to it. If one of these conditions are true, new intersection points and new lines are generated in the area such that boundary lines for future areas (not yet generated) that split up the old area along the intersection will exist. The procedures (line-line and line-area) are repeated until no new intersections can be found. Finally, as presented in section 4, to find relevant new areas, pattern recognition is used to find all areas from the intersection points within an original area, and then only the smallest new areas within the original areas are subject to output. Section 5 presents a C++ program and some examples that test the method's correctness and efficiency. At the end, conclusions and a research outlook will be given in section 6.

1.4 *Data model*

In Figure 3 a data model in EXPRESS-G (Schenck & Wilson 1994) is shown as background for the method to be presented. A three dimensional "Point" is defined that can be projected on a two dimensional surface as a "Point (2D)". A "Line" that is a line of the geometrical components, consists of two "Point"s. Note that a line may be a "Column" as will be explained in section 2. An "Area" that is an area of the geometrical components, consists of four "Line"s. For procedural reasons, "Point", "Line", and "Area" are stored in sets. Furthermore, a single "Area" can be coupled to other "Area"s using a table defined as "Area table".

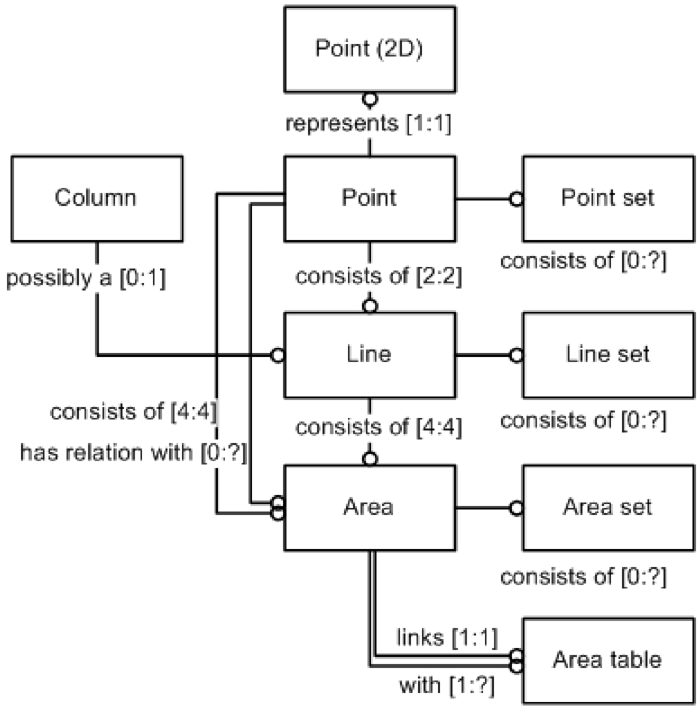

Figure 3. Data model.

2 LINE-LINE INTERSECTIONS

Geometrical components consists of points and lines that are defined by the points. These lines either define areas or define just the line itself. To distinguish between these two situations, only lines that define just the line itself are marked as "Column". All points and lines are stored in sets as shown in Figure 3, even if they are coincident, but in the latter case no new identity is assigned. However, if areas are coincident a new identity is used though. The different approach for points and lines and areas is related to the set-up of the spatial design and is not explained here. In all cases, lines and areas are defined by the first points and first lines with the relevant identity.

The line-line intersection procedure is the first procedure as part of the method presented in this paper. It contains of two steps. First, it is investigated, for a combination of two lines, and only if a line is marked as "Column", is partly or fully collinear with another line (which may be a "Column" as well). If this is the case, additional lines are generated as shown in Figure 4. In all other situations, not shown in figure 4, no action is taken. For instance, for the situation where lines join both end-points or join a single end-point and no other points. The same is valid when a column is fully enclosed but shorter than a line that is not a column. The reasoning behind this first step is that the general line-line intersection step, to be presented, does not include a solution for collinear lines. This is not problem as far as lines that define areas are processed, but for lines actually defining the geometry itself (columns), redefinition should be taken into account.

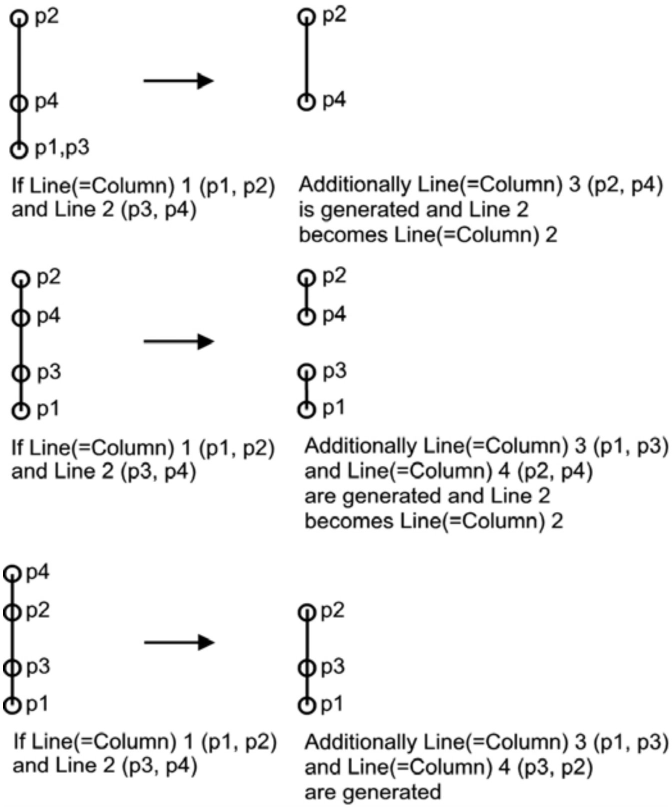

Figure 4. Column-Line collinear intersection.

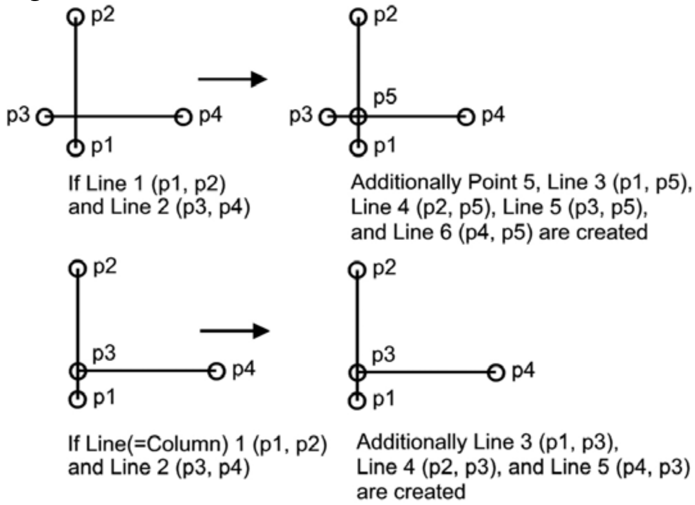

Figure 5. General line-line intersection.

The second step, general line-line intersection, checks every line-line combination, regardless whether the lines are a column or not, for intersection. The intersection is found by first checking whether the lines are positioned in a common plane. If this is the case, the line points are defined as "Point (2D)"s, as shown in figure 3, whereafter a possible intersection point is found in the specific two dimensional plane. If indeed an intersection exists, new points and lines are created as shown in figure 5. If two lines only have a single end-point in common, no action is taken.

The two steps are repeated for each line-line combination once. This does not guarantee that for now all lines are redefined completely, because during the process previous processed lines are not compared against the newly created lines. For practical situations however, a large part of the problem regarding the lines is solved and continuing with the next procedure (line-area interaction) seems to be advisable. Also because during this next procedure new lines are created, thus making a few more line-line intersection procedures needed anyway.

3 LINE-AREA INTERSECTIONS

After line-line intersections, now line-area intersection are investigated via a second procedure. For each line-area combination, three situations can be distinguished. First of all, it is possible that a line (being either a column or a line as part of another area) intersects with a line of the area considered. This is, giving the orthogonal limitation of the problem, possible for a line in plane with the area or a line perpendicular to the area. In both cases likewise, though, additional points and one new line are created as shown in figure 6 at the top: The intersection point of both lines is projected at the opposite area line and a line is defined between the intersection point (p7) and the projected intersection point (p8). Both points (p7 and p8) are marked as being "intersecting points" for the area, the relevance of this will be shown in section 4. It should be noted that other lines (those that will define the remaining lines of the two new and smaller areas) are not yet defined. However, these will be created in a next iteration where again line-line intersection will be investigated (see at the end of this section).

The second situation describes a line having one or two end-points in the area. This is the case for a line in-plane and completely within the area (second row figure 6) or a line perpendicular to the plane with only a single end-point touching the area's surface (third row figure 6). For this case, additional points and lines are created such that new areas can be found (in a next step, presented in chapter 4) that split up the original area appropriate. Points p5 to p12 (second row) or p5 to p11 (third row) are again defined as "intersecting points".

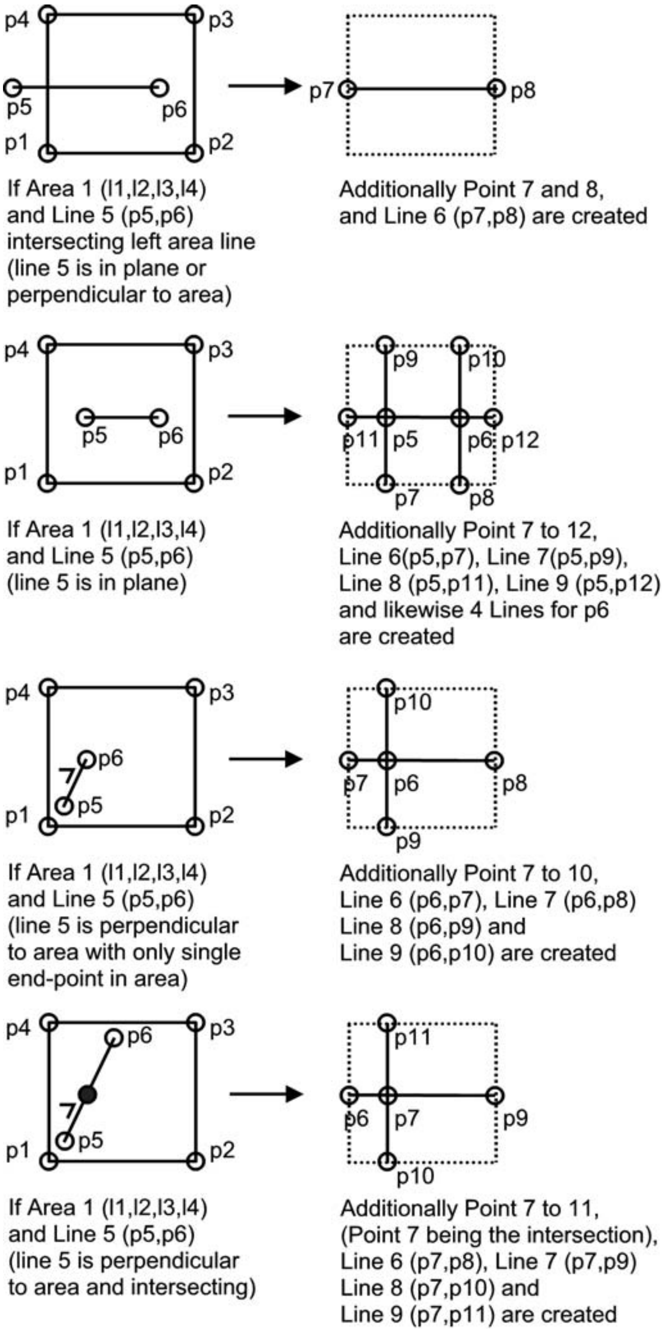

Figure 6. Line-area intersection.

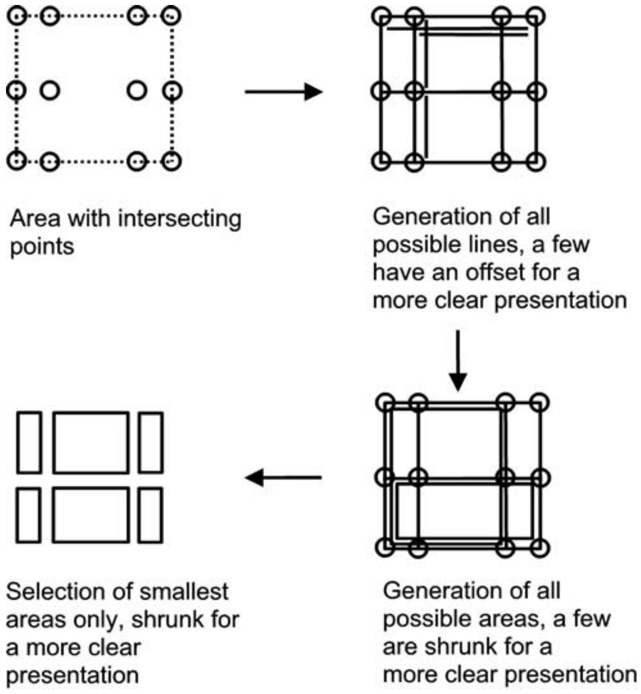

Figure 7. Generation of new areas.

The last possible situation is that of a line perpendicular to the area and intersecting, as shown in the fourth row of figure 6. Now five (also "intersecting") points are created (because the intersection does not exist already) and four line in order to create new areas in a next step (see section 4).

For all situations in section 2 and 3 if a point or line to be created is found to be already existent, the creation will not be carried out.

The procedures line-line interaction in section 2 and line-area interaction in this section are executed iteratively until no new points and lines are found. This assures that the problem is completely solved, at least for the intersection of lines and existing areas. In the next section, the new areas to be created will be taken into account.

4 AREA GENERATION

The aim of the approach presented in this paper is to redefine (by splitting) lines and areas such that they can easily be meshed as shown in figure 1. In the previous two sections, lines and areas have been split but no new areas, which redefine the existing areas, have been defined. The latter problem is presented in this section.

The procedure starts with, for each existing area, generating all possible lines for all "intersecting point"-combinations as shown in figure 7 at the top. Note that many lines overlap and are not visible in the figure, however, to make the principle clear, a few lines are given an offset, thus becoming visible. Hereafter all possible areas for all these lines are generated, as also shown in the figure. Again, many areas have overlapping lines and are thus not visible. For this reason, a few areas are drawn slightly shrunk. All the areas are stored in a set that is related to the area under investigation using a table, Figure 3. If the area under investigation does not contain any other areas, it is not needed to process the area further and it will be directed to the output.

A last step within this procedure is to find only the new areas that do not contain intersection points, except the four corner points. These areas are thus also the smallest areas out of which the original area can be build, figure 7 at the left bottom.

Finally, some selecting process is needed for the lines and areas that are part of the solution. Lines that are defined as "columns", and do not join collinear lines within, are defined as part of the solution, including their end points. Then, the input areas that did not contain other areas initially and the smallest areas found within the other input areas are also used for further processing, including all their corner points.

5 C++ PROGRAM AND EXAMPLES

To verify the method presented and to test its efficiency, a C++ program and some number of examples are presented in this section.

The problem as presented in this paper is, as discussed in the introduction, part of a larger so-called "research engine". And this research engine is under development using the programming language C++ via the Eclipse Platform version 3.4.1, Cygwin version 1.90.4.1, and Microsoft's Windows XP. For the problem in this paper thus a C++ program was written as well, using the normal object-oriented principles, and the data model as shown in figure 1. Because the procedures presented in section 2 to 4 were programmed exactly equivalent to the procedures as described, no code or code-extracts will be presented here.

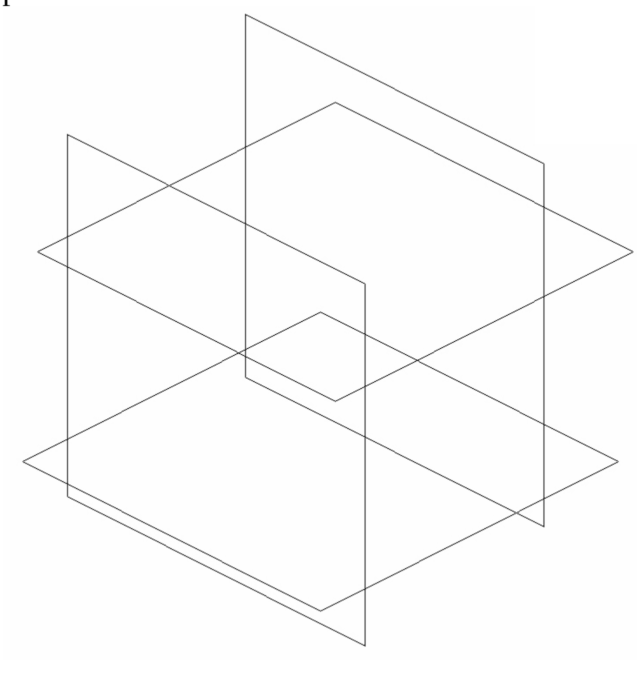

Figure 8. Input of an academic problem having 4 areas.

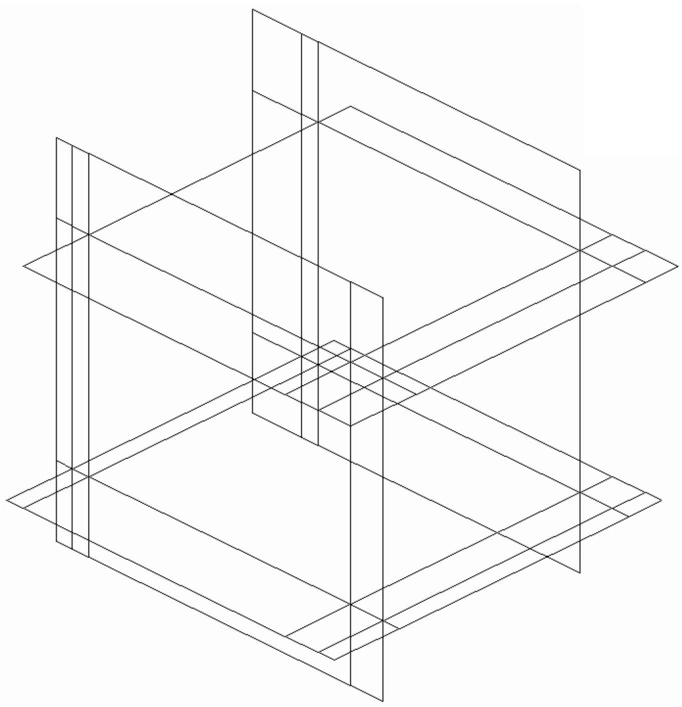

Figure 9. Output of an academic problem having 4 areas.

To demonstrate the solution method in this paper, first two academic problems will be presented with their input and output of the program.

To start with, in figure 8 an initial problem with four orthogonal areas is shown. For clarity only the areas' boundary lines are drawn, and in the design the area's are deliberately positioned relatively to each other with small offsets such that each area influences all other areas. After the procedures have been applied, the result is shown in figure 9. All areas have been split correctly. It may occur in first instance that too many split actions have taken place, for instance at the figure's left top corner, but it should be noted that the lines associated with the split areas come from somewhere else in the three-dimensional problem. To put it differently: All area cutting lines should be continued in their length direction along the complete problem geometry, which is clearly the case in figure 9.

In order to test the program for additional lines and columns (being lines to be kept in the output), a problem was defined as shown in figure 10. A single vertical column intersects two areas and a single line intersects two areas horizontally. Line and column intersect somewhere in the middle of problem's geometry. The solution is shown in figure 11 and here it should be realised that the column remains in the solution, however, split in four parts. One part is above the highest positioned area, one part between this area and the (former) horizontal line, etc. Note that the areas are split as if the horizontal line still exist, although this line has removed from the output as it is no column. This can not be avoided due the procedures defined, but this is not a problem for only the number of correctly defined areas increases slightly.

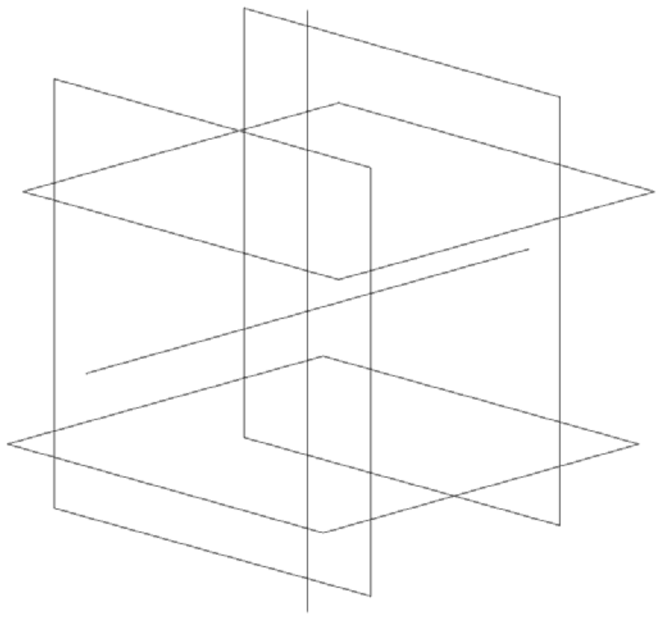

Figure 10. Input of an academic problem having 4 areas, a horizontal line and a column (vertically).

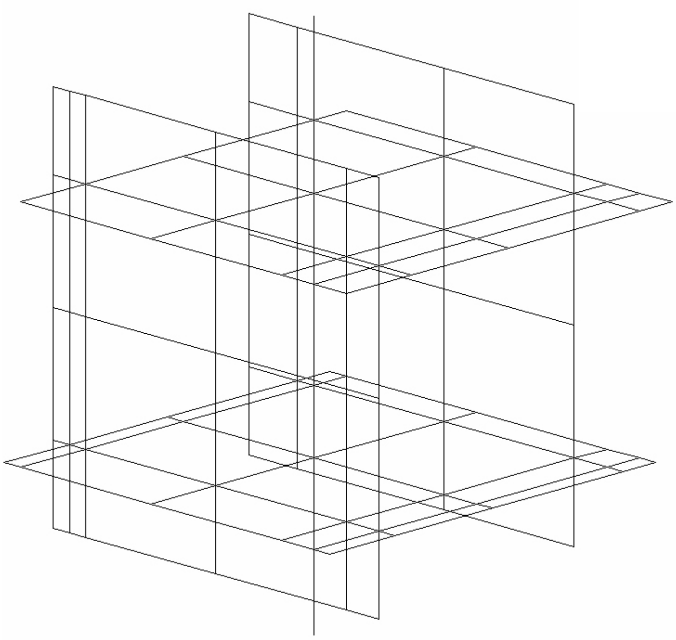

Figure 11. Output of an academic problem having 4 areas, a horizontal line and a column (vertically).

Figure 11 also shows clearly that as the problem gets more complex, it is increasingly difficult to present the problem and solution clearly by a static two dimensional projection. It was experienced that rotating the solution, using hiding techniques, and shrinking the areas was necessary to verify the program's procedures, figure 12.

The efficiency of the procedures and as such implemented in the program have been investigated as shown in figure 13. The testing started with an input of only lines, all being columns, and each line at least intersecting one other line. For an increasing number of lines, the calculation time undoubtedly will increase, however, the measurement in seconds did not reveal a difference as all situations, 2 to 5 lines, were executed in less than a second. For an input of only areas, with all areas intersecting at least one other area, execution time increases very aggressively and 50 seconds on a modern personal computer for 5 areas seems not to indicate a very efficient procedure. Giving a combination of lines and areas, for which every element intersects at least one other element, execution time is in between the two limit cases of only lines and only areas.

An influencing aspect of the procedures on the execution time is that for line-line and line-area combinations all possible combinations are investigated and that searching for inner areas in existing areas is carried out using all possible combinations of intersection points (see section 4).

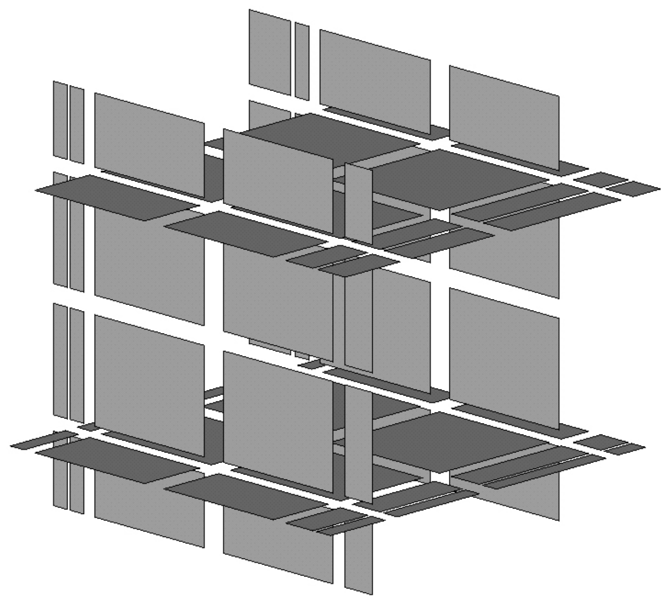

Figure 12. Output of the problem using hidden line and shrinking options (vertical column is not shown as only areas are drawn)

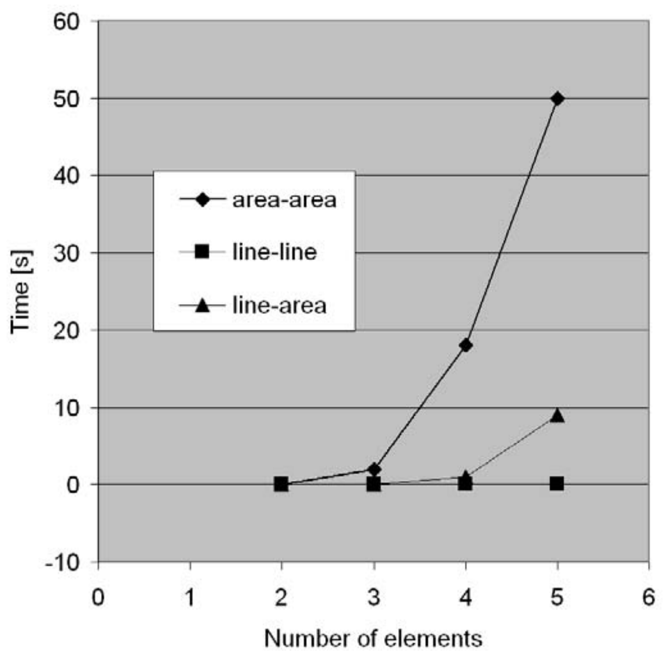

Figure 13. Execution time for several combinations of lines and areas.

A procedure could be developed that relates each line or area to a limited group of elements (lines and areas) in distance near to the line or area under investigation. Then only combinations are investigated within the group. In this way not the problem of combinatorial explosion is fundamentally solved, but the strong reduction in element numbers will yield practical building designs solvable. For the problem of inner areas, it is also possible to group promising intersection point combinations by means of a geometrically related reducer, as already used for three-dimensional pattern recognition and explained in (Hofmeyer and Kerstens 2006).

Another influencing factor on the execution time is the iterative sequence of line-line and line-area procedures. Instead of the current design, it may be more efficient to let the line-line intersection converge first, before starting line-area intersection. To verify this the interaction and convergence of these two procedure types is difficult to generalize for all possible designs and therefore it is probably the best to carry out some case-studies to find the most optimal sequence.

6 CONCLUSIONS & OUTLOOK

Procedures have been presented that enable the redefinition of geometrical components (lines and areas) such that they can be used as input for a finite element procedure to specify kinematically undetermined behaviour. All these procedures are part of a future research-engine that enables the research of building design processes.

The procedures consist of iteratively checking line-line and line-area intersections and finding inner areas in existing areas, three-dimensional but limited to orthogonal positions of the elements.

A C++ program has been used to successfully verify the procedures for academic examples. It has been shown that even for small scale examples, some care is needed in presenting the results clearly (hidden lines and shrinking techniques). Although the examples are simple, it can be understood by observing the solutions that processing them manually is very difficult.

Calculation time increases more strongly for increasing area numbers than for line numbers but some improvements have been suggested to overcome this problem.

In the near future, the procedures presented may be generalised for non-orthogonal positions of the elements, for bodies, and even for curved components. Furthermore, the procedures can be made more efficient, can be compared with other strategies to redefine the components, and should be implemented in the research-engine mentioned.

REFERENCES

Aus, J. et al. 1996. *How to-Integrate CAD and Analysis*. Glasgow: NAFEMS.

Bletzinger K.U., Ramm E. 2001. Structural optimization and form finding of light weight structures. *Computers & Structures* 79(22-25): 2053-2062.

Eastman C., Jeng T.S. 1999. A database supporting evolutionary product model development for design. *Automation in construction* 8(3): 305-323.

Fenves S., Flemming U., Hendrickson C., Maher M.L., Quadrel R., Terk M., Woodbury R. 1994. *Concurrent Computer-Integrated Building Design*. Englewood Cliffs, New Jersey: PTR Prentice Hall.

Fenves S.J., Rivard H., Gomez N. 2000. SEED-Config: a tool for conceptual structural design in a collaborative building design environment. *Artificial Intelligence in Engineering* 14: 233-247.

Frey, P.J., George, P.-L. 2000. *Mesh Generation, application to finite elements*. Oxford: Hermes.

Haymaker J., Fischer M., Kunz J., Suter B. 2004. Engineering test cases to motivate the formalization of an aec project model as a directed acyclic graph of views and dependencies. *ITcon 9*: 419-441.

Hofmeyer, H. & Bakker, M.C.M. 2008. Spatial to kinematically determined structural transformations. *Advanced Engineering Informatics* 22(3): 393-409.

Hofmeyer, H. & Kerstens, J.G.M. 2006. Full 3D Structural Zoning of Space using a Geometrically Related Reducer and Matrix Coupling. In Kaga A., Naka, R. (eds), *CAADRIA '06, Proceedings of the 11th Conference on Computer Aided Architectural Design Research in Asia, March 31-April 2, Kumamoto, Japan*: 161-168. Kumamoto, Japan: Kumamoto University.

Hofmeyer, H. & Kerstens, J.G.M. 2008. A bird's-eye view of cyclic spatial-structural transformations, grammar based design, and structural optimization. In M. Muylle (ed.), *eCAADe 2008, Architecture 'in computro', Integrating methods and techniques, Proceedings of the 26th Conference on Education and Research in Computer Aided Architectural Design in Europe, September 17-19, 2008:* 483-492. Antwerp, Belgium: eCAADe and Artesis.

Hofmeyer, H. & Russell, P. 2009. Interaction between spatial and structural building design: A finite element based program for the analysis of kinematically indeterminable structures. In *CONVR2009, 9th International Conference on Construction Applications of Virtual Reality 2009, November 5-6, 2009*.

Hofmeyer, H. 2007. Cyclic application of transformations using scales for spatially or structurally determined design. *Automation in Construction* 16(1): 664-673.

Khemlani L., Timerman A., Benne B., Kalay YE. 1998. Intelligent representation for computer-aided building design. *Automation in Construction* 8(1): 49-71.

Knupp, P., Steinberg, S. 1993. *The fundamentals of grid generation*. Boca Raton: CRC Press.

Kocaturk, T., Van Weeren, C., Kamerling, M.W. 2003. Integrating Architectural and Structural Form-Finding Processes for Free-Form Digital Architectures. In: Bontempi, F. (ed.), *ISEC-02, System-based Vision for Strategic and Creative Design, Volume 1*: 557-562. Lisse, The Netherlands: Balkema.

Kotsopoulos S.D. 2006. *Constructing Design Concepts, A Computational Approach to the Synthesis of Architectural Form*. USA: Massachusetts Institute of Technology (MIT).

Matthews K., Duff S., Corner D. 1998. A model for integrated spatial and structural design of buildings. In: Sasada T., Yamaguchi S., Morozumi M., Kaga A., Homma R. (eds), *CAADRIA '98: Proceedings of The Third Conference on*

Computer Aided Architectural Design Research in Asia.: 123-132. Osaka: Osaka University Japan.

Mora R., Rivard H., Bédard C. 2006. Computer representation to support conceptual structural design within a building architectural context. *Journal of Computing in Civil Engineering* 20: 76-87.

Rafiq M.Y., MacLeod I.A. 1998. Automatic structural component definition from a spatial geometry model. *Engineering Structures* 10(1): 37-40.

Rafiq, M.Y., Mathews, J.D., Bullock, G.N. 2003. Conceptual Building Design-Evolutionary Approach. *Journal of computing in civil engineering* 17(3): 150-158.

Rivard H., Fenves S.J. 2000. A representation for conceptual design of buildings. *Journal of Computing in Civil Engineering* 14: 151-159.

Van Roosmalen, M. 2008. *Honours Program Research TU/e*. Honours Programme report of Eindhoven University of Technology, The Netherlands.

Rottke, E. 1998. *ExTraCAD, Computerunterstützung des architektonischen Traagwerkentwurfs*. Aachen, Germany: RWTH.

Scherer R.J., Gehre A. 2000. An approach to a knowledge-based design assistant system for conceptual structural system design. In: Goncalves, Stegier-Garcao, Scherer (eds), *Product and Process Modelling in Building and Construction*. Rotterdam: Balkema.

Thomson, J.F., Warsi, Z.U.A., Mastin, C.W. 1985. *Numerical grids generation, foundations and applications*. Netherlands: North Holland.

Schenck D., Wilson P. 1994. *Information Modeling the EXPRESS way*. New York, Oxford: Oxford University Press.