Automation and Robotics in Construction XI
D.A. Chamberlain (Editor)

Computer-Aided Planning in Masonry Construction

H. Abdul-sater and A. Delchambre

CRIF - Research Centre for the Belgian Metalworking Industry, Industrial Automation Department
CP 106 - P4, 50, Av. F.-D. Roosevelt, B-1050 Brussels, Belgium
Email: ALAIN__DELCHAMBRE_CRIF@eurokom.ie

Abstract

Nowadays, robotic systems are considered as key technologies for construction automation and a way to increase the productivity of the building process. The efficiency of the robot work could be increased by a suitable high level planning system, whereby the sequence of robot working points, the wall's sequence and the stone flow are computed off-line. Moreover, a global simulation is a means to validate the pre-computed sequences and allow the user to make possible modifications.

This paper presents such a high level planning system. In this system, walls are partitioned into regular and irregular stones, starting from the architect drawings. Irregular stones are cut and grouped on pallets at the sawing factory. All the robot working positions, pallet compositions, positions and flow, the wall and stone sequence are computed off-line.

1. INTRODUCTION

The building process problems of the construction industry can only be solved by a long term strategy of flexible automation and integrated data and information processing. It is necessary to build up an integrated system, which embraces all aspects of the construction process, manipulator systems, control technology and planning and programming tools.

By the integration of the entire building process from architectural planning, computer aided generation and simulation of robot programs to the automated wall assembly of bricks or large formatted stones, rationalisation can be achieved without limiting the individual planning freedom of the architect.

The first efforts in rationalisation are made by enlarging the format of bricks, to speed up the process. In addition, sawing equipment has been developed because the larger stones were too expensive and too difficult to cut accurately by hand. This development has continued until today by improving the accuracy of the stones, and by improving the logistics, so that the stones can be automatically pre-cut in the sawing factory instead of on the construction site. Moreover, walls are partitioned into standard and non standard stones. The number of standard stones and the dimensions of each non standard stone are pre-computed according to an optimising strategy.

The productivity could be improved, furthermore, by automating the process and using a robot system. This requires a highly flexible and powerful programming system that includes generation of the robot programs (vehicle and manipulator) and task planning.

No repetitive robot work is possible because each assembly task is different from the previous. A transformation of the geometric data of an architectural plan to the motion information of the manipulator and vehicle is then necessary. This work can be divided into 5 components:

1- the determination of the position and shape of stones in the wall;
2- the determination of the position of stones on the pallets;
3- the determination of the position of the pallets and of the robot system on the floor;
4- the determination of the vehicle path between the robot positions;
5- the determination of the manipulator path between the pallet and the wall.

The following sections describe the **palletising problem** (point 2) and the **task planning** (points 3 and 4). They mainly focus on the different strategies considered to group stones on pallets, and conclude by showing some results.

2. SYSTEM FUNCTIONS

Figure 1 presents the global architecture of a robot assembly system for computer integrated construction.

This system includes two esential parts :

(i) an on-line part;

(ii) an off-line-part.

Nevertheless, only the off-line part is detailed on figure 1.

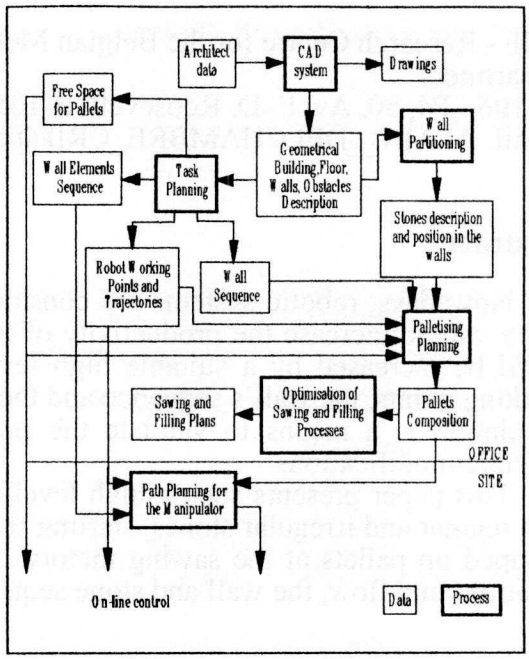

Fig1: off-line programming structure.

2.1. On-line programming

The on-line programming system controls the execution of the vehicle and manipulator movements in order to guarantee the obstacle-free paths computed off-line. It includes four main functions:

(i)vehicle motion supervision and error diagnosis;

(ii) vehicle motion error recovery;

(iii)manipulator motion supervision and error diagnosis;

(iv) manipulator motion error recovery.

2.2. Off-line programming

The heart of the off-line programming is a database. All programming modules are connected to the database where they exchange different data. The off-line programming is divided into two major parts:

(i) work executed at the office;

(ii) work executed at the construction site.

2.2.1. Off-line programming at the office

It includes four main functions:

(i) wall partitioning;

(ii) task planning;

(ii) palletising planning;

(iv) optimisation of sawing and filling process.

2.2.1.1. Wall partitioning

Given the architect drawings, each wall is partitioned into *regular* and *irregular* stones according to optimising critiria. For example:

(i) mimising the overall number of stones in a wall;

(ii) mimising the surface of cuts;

(iii) reducing as many stones as possible of identical shape.

2.2.1.2. Task planning

Given the robot description and reachability, the walls and obstacles description and position, and a starting point (the wall to be first built), the task planning process consists in finding the sequence between walls that minimises the global trajectory of the robot on the construction site. However, this must also insure that the robot will not be blocked by the walls it has built. Refering to the figure 2.,where the position of the numbers corresponds to the robot position with regard to the wall, it is easy to conclude that the wall sequence 'A' in this example is possible, however, the sequence 'B' is impossible.

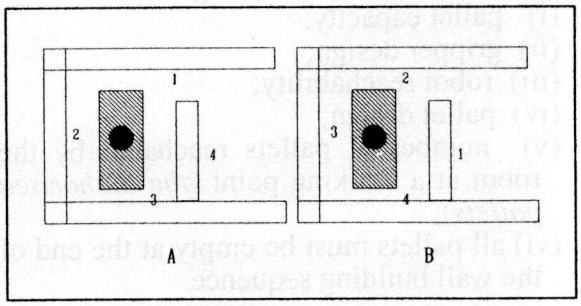

Fig. 2: Wall sequence.

The task planning process delivers the sequence of wall elements, a list of robot working points, the free space for pallets positioning arround each of them, and finally the obstacle free trajectories for vehicle motion.

2.2.1.3. Palletising planning

The wall partitioning and sawing process are based on an optimising strategy. Cutting the stones at the sawing factory instead of on the construction site, however, raises the additional problem of how to transfer the stones from the factory to the construction site. The classical way for the transportation of stones consists of grouping them on pallets, which is straightforward if the stone length is uniform. Problems arise when stones have different sizes. The main problem is how to group stones on pallets in order to reduce the pallet count with respect to several constraints e.g.'s stones of different walls can not be located on the same pallet, the pallet design and capacity, and the gripper design. This is the **palletising problem.**

Palletising planning delivers a list of pallets and the composition of each of them.

2.2.1.4. Optimisation of sawing and filling process

Wall partitioning delivers a list of stone descriptions. Each description element belongs to a single stone. It contains data about the stone dimensions, and the pallet onto which a certain stone has to be placed.

The first, straightforward, way for sawing would be to produce the stones for a pallet one after the other. But this simple procedure also results in much waste. The optimum solution would be to produce the stones for all pallets in parallel. But there are only a limited number of pallet positions available at the exit of the sawing machine. Therefore, a separate task, independent of wall partitioning, optimises the sawing process with respect to the available pallet positions at the exit of the sawing machine.

2.2.2. Off-line programming at the construction site

The main function of off-line programming at the construction site is to plan paths for the manipulator movements i. e. between the pallets and the walls.

3. THE PALLETISING PROBLEM

3.1. Problem description

The figure 3 shows the lay-out of a partitioned wall where the labelled stones are the *irregular* ones. It shows also the robot at a pre-computed working point.

Given the robot's reachability and technical constraints, at each working position the robot is able to build only a part of a wall (this we call 'sub-wall') which contains a certain number of irregular stones, regular stones, and/or other elements like *lintels,* for example. *Irregular stones*[1] belonging to this part of wall must be grouped on pallets according to an optimising criterion as well as to some constraints. The optimising criterion is the reduction of pallets to be used while

the constraints are the following ones:

(i) pallet capacity;
(ii) gripper design;
(iii) robot reachability;
(iv) pallet design;
(v) number of pallets reachable by the robot at a working point (*the authorised pallets*);
(vi) all pallets must be empty at the end of the wall building sequence.

Furthermore, we assume that stones of different walls (even adjacent walls) can not be mixed on the same pallet. This is realistic due to the non uniformity of material and thickness among the walls.

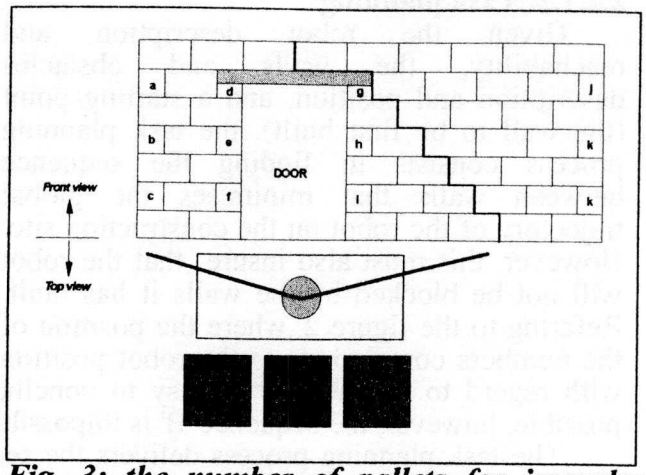

Fig. 3: the number of pallets for irregular stones is 3.

Finally, the number of authorised pallets is not uniform for all robot working positions. It depends on the *free space* at this position.

3.2 The gripper design and the palletising modes.

The gripper design introduces a hard constraint that can completely modify the stones arrangement on the pallets, i.e. the palletising mode. If we consider a gripper able to handle stones from their *upper* side, stones can then be *mixed* on a pallet; in this case, all stones are reachable by the robot at any time. However, if the gripper must handle stones from a *lateral* side, then at each time the stone to be handled must be located at the *outer* side of the pallet (see fig. 4 where the shaded area represents the empty space on the pallet and the numbers represent a possible sequence following which stones will be handled)

If we consider that the number of authorised pallets for the current working point is $\mathbf{N_{Pa}}$[2] (3 in the case of figure 3). Two palletising mode could be defined as following:

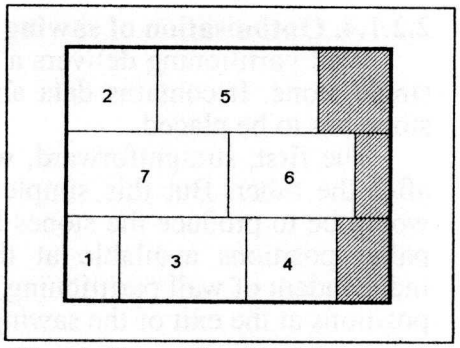

Fig. 4: Second palletising mode.

[1]Only the irregular stones are considered in the palletising process.
[2]The mark "a" means *authorised.*

MODE 1 : the irregular stones could be mixed on at most $\mathbf{N_{Pa}}$ pallets at a time and have to belong to the current sub-wall.
MODE 2 : same conditions as in mode 1 but the requested stone must be at the outer side of one of the $\mathbf{N_{Pa}}$ pallets (due to the gripper design).

Consider now that a pallet may contain stones belonging to the *next* sub-wall. In this case, after building the current sub-wall, some stones could remain on some pallets. This introduces two additional palletising modes:

MODE 3 : the stones of the current and next sub-wall are considered and could be mixed on the present pallets.
MODE 4 : same conditions as in mode 3 but the requested stone must be at the outer side of one of the $\mathbf{N_{Pa}}$ pallets.

3.3. The optimising algorithm

3.3.1 The grouping problem in general

A number of problems involve grouping elements of a set into a small number of families. These are the typical problems of classification, economy of scale, Group Technology, Bin Packing, Line Balancing, Shop Layouting and the like.

A closer look at the palletising problem reveals some similarities between this problem and the Bin Packing problem. It is the problem of grouping unidimensional objects (stones of different lengths and the same thickness) together in such a way that no group's total size exceeds a given constant (a *row* length), and to use as few groups (rows) as possible for all the objects.

The big difference that exists between both of these problems is that the objective of bin packing is minimising the *rows,* while for palletising the objective is to minimise the *pallets*, i. e. the number of *row groups*. This means that one approach for solving the palletising problem is to solve the bin packing problem beforehand.

3.3.2 Grouping and palletising

The optimising algorithm used for palletising is composed of two parts :

(i) A *Grouping Genetic Algorithm,* GGA. Given a list of stones of the same thickness, their length and a maximal *Row length* (i.e. the pallet length Lp), this algorithm gives the best arrangement of stones, i.e. the one that minimises the rows count.

(ii) The *Palletising Algorithm,* PA. This uses the GGA to create pallets according to the selected arrangement mode.

The reader not familiar with the Genetic Algorithm paradigm (GA) should consult the literature on the topic (e.g. [Morrow 91]) as space does not allow its introduction here [Goldberg,89], for instance, offers an excellent presentation of much of the GA technique, while [Falkenauer and Delchambre,91] present a GGA for the Line Balancing and Bin Packing problems.

Let us, however, recall that the main concept underlying the GA mechanics are *schemata* - portions of chromosomes which map onto subspaces of the search space. In fact, the efficiency of GA's (or, more precisely, of the crossover operator) stems from the fact that the algorithms, while manipulating the chromosomes, implicitely manipulate large numbers of schemata ([Holland,75]).

Note that this two-stage approach (first grouping stones in rows and then grouping rows in pallets) could in principle be replaced by one where whole *pallets* are considered instead of *rows*. However, that would require the GA to work on *two-dimensional* structures subject to rather ill-defined precedence constraints, a problem for which the current state of the art does not offer a conclusive solution.

3.3.3. Palletising algorithm

Assuming that the GGA guarantees the optimal grouping of stones in rows, the remaining problem is still how to group rows in order to build pallets with respect to the objectives already mentioned in section 3.1. Different approachs could be considereded. The one that fulfill all the objectives of the palletising problem is a *recursive* approach. It consists of the following : Given a list of stones as a primary list (**PL**) to be palletised by order of their assembly sequence

START: { group the stones simply by using the GGA;
IF the number of the rows obtained is higher than the maximal authorised[3] at the current working point
THEN { extract the **last stone** in the sequence from the list;
GOTO START;
}
ELSE a rows group is formed;
}
This operation has to be repeated with the remainder stones;

This ensures that the size (in term of number of rows) of the first group of rows, not only matches the authorised size, but also that stones in this group form a consecutive list which is part of the PL.

It is clear that this method will always find a satisfactory solution, but it is highly time consuming. Some improvement, in terms of searching time, an be added to this method. First by considering only a part of the primary list at a time and this by means of an ***estimation***. Second, by the introduction of two ***heuristics***. Let us describe this estimation and heuristics in the order of their application.

a) The heuristic HE1 consists of :

1- order the stones into the increasing order of their assembly sequence;
2- take the first stone and put it in the first row;
3- add stones to this row until the row length (pallet length) is reached, then take a new row;
4- the number of total rows that is reached is called "N_{Rh}".

Thus, before running the GGA, we compute the N_{Rh} (in rows) value for the whole wall. The GGA will be run only if the following condition **C** is performed:

$0 <$ REMAINDER OF $(HE1/N_{Rp}) \leq \alpha$. **(C)**

Where N_{Rp} is the number of rows per pallet and $0 < \alpha < N_{Rp}$.

If not, the grouping order (in rows) is the one given by HE1. Thus, pallets are produced by grouping the rows in the order in which they have been produced by the HE1 (first pallet = row1, row2 and row3. Second pallet = row4, row5 and row6, etc.). α is to be adjusted by means of tests on different instances of the treated problem. Also the condition C could be modified in order to take into account the filling ratio of rows. Let's consider, for example, $N_{Rp} = 3$ (pallet capacity = 3 rows) and $\alpha = 1$. The condition C means that the GGA will be run only in case of N_{Rh} takes one of the following values: 4, 7, 10, 13, 16, etc. If we consider that $\alpha = 2$, the GGA will be run only if N_{Rh} takes one of the following values: 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, etc.

In case of the GGA has to be run, the following estimation and heuristic are applied.

b) The estimation ES1: consists of taking, at a time, only a consecutive part of the primary list, a sub-list (**SL**). The total length of SL has to be inferior, but as close as possible, to the total length of the authorised rows.

c) The heuristic HE2: after running the GGA, if the difference between the number of rows obtained and the authorised number is n, for instance, we do the following :

1- we order the rows in the increasing order of the total length of stones in each row;
2- we consider then the first n rows and the total length (L) of stones they're onto ;
3- we extract stones, starting by the bottom of SL and by considering their total length at each step. We stop when the length L is reached.

This heuristic seems to be satisfactory in many instances of the palletising problem. There could be some cases where after applying this heuristic, the desired result remains not reached. This is why the use of this heuristic only *speeds up* the search but cannot guarantee to find the expected result immediately. This recursive method is applicable for all palletising modes nevertheless, there are some particularities for modes 2 and 4. For these modes we must consider a *variable* value for N_{Rp} and consecutively N_{Ra} according to the position of the current stone on the pallet. Consider the hypothesis used above (N_{Rp}=3 and N_{Pa}=3), this means that N_{Rp} will take two different values, respectively 2 and 1. Consecutively, N_{Ra} will take the two values, 6 (N_{Rp} x 2) than 3 (N_{Rp} x 1). This insures that first stones of the considered sub-list are at the outer side of the pallets and those of the end are at the inner side.

3.3.4 Palletising tests

Consider table 1, it contains a short description of 29 irregular stones. For each stone, it shows its label, length, the sub-wall (or working point) it belongs to and its order in the assembly sequence. The tables 2, 3, 4 and 5 present the palletising results using the different modes with a pallet length L_P = 1000 mm, the number of authorised pallets = 3 and each pallet contains at most 3 rows.

3.3.5. Discussion

Looking to the tables, one can verify that they all respect the corresponding palletising mode. The first interesting comparison to be made is between the row and pallet numbers deduced from each table, and those computed by the GGA regardless to any palletising mode. Computing the corresponding *optimal* row and pallet numbers we obtain cosecutively **16 Rows** and **6 Pallets** (the last pallet contains only one row !). In term of row numbers, the results of modes 1 and 3 are quite similar to the optimal one while both of modes 2 and 4 require more rows. In term of pallets number, all modes, except mode 2, are similar to the optimal one. This is because mode 2 introduces the most severe constraints for grouping and reduces consequently the possible optimisations. Thus the difference between the *less* and the *most* optimal arrangements is (for this example) one pallet only.

Consider now that a pallet could contain 4 rows instead of 3. The required pallet numbers for the different modes is consecutively: **4, 5, 4 and 5,** while the *optimal* result is **4** pallets.
The second comparison that could be made is between the row and pallet numbers obtained by the general grouping approach and those obtained by the heuristic HE1, i. e, by computing the N_{Rh} value and considering $\alpha = 1$. This leads to **18 Rows** and **6 Pallets.** In this case both results give the same number of pallets and the optimising procedure seems to be not necessary. This example is very important and needs more discussion. Let us consider a new set of stones composed of the current one, to which we add one stone of 100 mm length. The number of rows given by HE1 is now 19 and the number of pallets is consequently 7. While the result obtained by the palletising algorithm is still the same (i. e. 6). Indeed, in this case the condition C with $\alpha = 1$ (see section 3.3.3.a) is performed and implies the use of the optimising algorithm.

4. CONCLUSIONS

This paper has shown the structure of a high level planning system for computer integrated construction. It mainly mentioned the role of the off-line programming for the grouping of irregular stones which contributes to a fully automated sawing and filling process. Regarding the robotised construction and especially the gripper design, four palletising modes have been developed and some results have been discussed. For our application, and due to the shape of stones we use (they present several holes at their upper side), neither palletising mode 1 nor 3 could be used. The palletising mode we have actually chosen is the fourth one because it allows more optimisation than mode 3.
The next tasks we are already working on are the task planning and the interface to a robotics-oriented CAD/CAM environment (ROBCAD for instance), not only to generate simulations, but also to create a feed-back to the task planning to modify, for example, some robot trajetories.

5. ACKNOWLEDGEMENTS

This paper describes a research done at the *Industrial Automation Department* of the CRIF/WTCM. Suport was provided in part by the EEC (ESPRIT III Project 6450 ROCCO).

REFERENCES

[Falkenauer and Delchambre,91] Falkenauer Emanuel and Delchambre Alain:
A Genetic Algorithm for Bin Packing and Line Balancing.

[Goldberg,89] Goldberg David E.:
Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wessley Publishing Company, Inc.

[Holland,75] Holland John H.:
Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.
[Morrow, 91] Morrow M.
Genetic Algorithms - A new class of searching algorithms, Dr.Dobb's journal,April 91.

(The empty line in the tables is used to separate the groups of pallets. This means that for table 2, for instance, the first group of pallets is composed of pallets1, 2 and 3, the second one of pallet 4 and the third one of pallets 5 and 6).

Stone Id.	Stone order	W. P.	Stone length (mm)	Stone Id.	Stone order	W. P.	Stone length (mm)
M	0	1	495	X1	15	1	248
G	1	1	684	X2	16	1	248
K	2	1	502	U	17	1	197
L	3	1	496	T	18	1	301
D	4	1	850	X3	19	1	248
Q1	5	1	329	V	20	1	498
R	6	1	322	N	21	2	489
C	7	1	857	E1	22	2	794
Q2	8	1	329	I	23	2	543
H	9	1	669	A	24	2	946
F	10	1	731	E2	25	2	794
O	11	1	448	X4	26	2	248
W	12	1	388	S	27	2	307
Y	13	1	108	B	28	2	932
P	14	1	431				

Table 1: Stones data.

W.P Id.	PALLET Id.	Stones in row 1	Stones in row 2	Stones in row 3
1	1	10, 15	1, 18	5, 9
1	2	0, 14	6, 12, 16	7, 13
1	3	8, 11, 17	4	2, 3
1	4	19, 20		
2	5	22	25	28
2	6	23, 27	24	21, 26
Rows number : 16			**Pallets number : 6**	

Table 2: palletising results for mode 1.

W.P Id.	PALLET Id.	Stones in row 1	Stones in row 2	Stones in row 3
1	1	0	9	4
1	2	1	10	5, 6, 8
1	3	2, 3	11, 12, 13	7
1	4	14, 15	20	16,17, 18, 19
2	5	22	25	24
2	6	23	26, 27	21
2	7	28		
Rows number : 19			**Pallets number : 7**	

Table 3: palletising results for mode 2.

W.P Id.	PALLET Id.	Stones in row 1	Stones in row 2	Stones in row 3
1	1	10, 15	1, 18	5, 9
1	2	0, 14	6, 12, 16	7, 13
1	3	8, 11, 17	4	2, 3
1	4	19, 23	20, 21	22
2	5	24	25	26, 27
2	6	28		
Rows number : 16			**Pallets number : 6**	

Table 4: palletising results for mode 3.

W.P Id.	PALLET Id.	Stones in row 1	Stones in row 2	Stones in row 3
1	1	0	9	4
1	2	1	10	5, 6, 8
1	3	2, 3	11, 12, 13	7
1	4	14, 15	20, 21	16, 17, 18, 19
2	5	22	27	24
2	6	23, 26	28	25
ROWS NUMBER : 18			**PALLETS NUMBER : 6**	

Table 5: palletising results for mode 4.