Optimal Manipulation Trajectory and Laying Pattern Generation Algorithm for Handling Robot

S. H. Lee[1], S. N. Yu[2], S. J. Lim[2], B. G. Ryu[2] and C. S. Han[3]

[1]Master course, CIM & Robotics Lab., Department of Mechanical Engineering, Hanyang University, Seoul, Rep of Korea; PH (82)31-400-4062; FAX (82)31-406-6398; e-mail: hopezic@gmail.com
[2]Ph.D. Candidate, CIM & Robotics Lab., e-mail: hymecer@gmail.com
[3]Professor, CIM & Robotics Lab., Division of Mechanical and Information Management Engineering, Hanyang University, Ansan, Rep of Korea; PH (82)31-400-4062; FAX (82)31-406-6398; e-mail: cshan@hanyang.ac.kr

Abstract

The conventional brick handling such as road paving site, building construction site is allocated to the construction workers. Since these kinds of tasks are obviously labor-intensive and tedious, there have been many approaches to the automation at the construction site. However, the automation of the block-laying task still has several problems caused by the poor surroundings - inequable construction materials and uneven working conditions like various laying and paving patterns. Herein, this paper proposes an integrated mobile manipulator system operated by the optimal laying pattern and trajectory generation algorithm. The pattern generator is designed by the "Fast Algorithm" based on Steudel's algorithm; the trajectory generation algorithm is based on the "Overlap Method" which is a treatment skill for robot-surrounded obstacles. This study mainly shows the efficiencies of the proposed pattern and trajectory generation algorithm for the brick-laying task and the performance evaluation of the prototype system.

Keywords: Brick Laying Pattern Algorithm, Manipulation Trajectory Generation Algorithm, Brick-laying, Brick-paving, Mobile Robot

1. Introduction

Automation in the industrial field, in specific, factory automation has made good progress. Operators used to be included in a conventional manufacturing process with uniform working conditions - a formal production line. Automation outside the production line, however, has several limitations and difficulties in adapting to actual conditions, manual or semi-automatic machining tools are mostly used in modern industries. Automated machines or robots are starting to work with man on the jobsite, specifically in a construction site. Through many reports, brick laying and paving is the laborious task with harsh repetitive working conditions; they often cause fatal injuries. Miedema and Vink (1996a, 1996b) found that the highest workload is experienced by the bricklayer when the bricks are located 0~50cm above the work floor and the highest workload of the bricklayers' assistant is seen in loading and unloading process. [1] To solve these problems, Anliker (1988) developed one of the earliest prototypes of the semi-automated bricklaying masonry machine which is able to build pre-assembled brick walls up to 8 meters long. [2] And Pritschow (1996) proposed a brick laying robot which can operate such a picking bricks or blocks task on the construction site from prepared pallets, applying bonding material, and erecting brickwork with high accuracy and quality. [3] However, these studies commonly have several critical drawbacks. First, they did not consider the importance of an optimized brick-laying pattern generation; hence, the constructor should design the laying pattern of the wall or load separately and check the possibility of the robot to perform the laying task (Fig. 1). Second, they did not pay attention to the motion optimization of the robot arm based on the brick laying position and surrounding obstacles. Motion and trajectory optimization can increase the efficiency of the entire task. This paper defined the brick-laying task and a couple of assumptions for the system configurations. Next, the main pattern generation algorithm - the "Fast Algorithm" is introduced briefly. Then, the designed trajectory generation algorithm to travel between the gripping point (initial point) of picked and palletized point (target laying position) is explained; its performance is verified.

Fig. 1 Conventional method and proposed robotic automation method

2. Concept Design of the Proposed System

2.1 Definition of Target Task

Brick-laying task firstly spreads mortar on sides or ends of brick already laid. The end of the laid brick is buttered with mortar and shoved against the preceding brick, spreading mortar on both sides of closure brick. Laying the brick into position and squeezing the mortar to a width of 10mm, the surplus mortar is extruded from the bed. Then, it is scraped off with the trowel; the same process is repeated onto the next. Since this repetitive characteristic accompanies harmful effects to the workers, the goal of the research was set up for assist these task.

2.2 Configurations of the Proposed System

Generally, the performance of the robotic system and the efficiency of the associated task are determined by adequately sharing the entire task between human and robot, and setting the limitation of the role of the robot system. The aim of this study is to design an assistive robot system which will help construction workers with the most repetitive, and monotonous procedures of brick laying, but not substitute them. Due to many kinds of uncertain field conditions, this study specifically limits the role of the robot to carrying the piled blocks and delivering to the target position by a generated optimal pattern and trajectory. As shown in Fig. 2, in the brick-laying procedure, squeezing the mortar, tapping down the brick, and extruding the surplus mortar from the bed are carried out by the construction workers; it shows the predefined working procedure of each task using the brick laying robot while considering the design strategy. In this study, the unit pattern generation area is defined to use the optimal pattern generation algorithm in the brick- laying robot. The maximized pattern generation unit area for every single stop, considering the robot motion range, is shown in the ($L \times W$) area of Fig. 2. The junction point is the area which the robot cannot cover using its own motion range. Therefore, the bricks belonging to that area are laid by the human worker simultaneously, while the robot performs the brick-laying task. If the user can define the laying timing sequence of various types of bricks, the complex laying is possible. Fig. 3 shows the data process of the entire system. In the following chapter, the Fast algorithm for the brick-laying pattern generation as an initial point of the OLP simulator and real robot system is briefly introduced.

3. Optimal Pattern Generation Algorithm

3.1 Definition of the Fast Algorithm

The "Fast algorithm" gets similar processes with Steudel's algorithm in generating the initial four solution patterns [4]. In addition, Treatment 3 is adopted to apply the heuristic to the central hole in the following three methods, recursively, so as to remove the overlapped area (Fig. 4).

(1) The 1st method: the bricks are cut at the two horizontal edges of the overlapped area.

(2) The 2nd method: the bricks are cut at the two vertical edges.

(3) The 3rd method: the bricks are cut at the left vertical edge and the lower horizontal edge.

As there are not any considerations about all of the block sizes in this algorithm, computing time should be ensured in the whole process. The initial solutions of the first phase find the combination, and define 4 parameters as follows. There are three fundamental rules for the second phase, that I just for

(1) $\bar{a}$ *: When maximizing the length of the block and disposing the bricks lengthwise, the maximal possible number of bricks* $= 5l$.

(2) $\underline{a}$ *: When maximizing the length of the block and disposing the bricks lengthwise, the minimal possible number of bricks* $= 2l$.

(3) $\bar{b}$ *: When maximizing the width of the block and disposing the bricks lengthwise, the maximal possible number of bricks* $= 8w$.

(4) $\underline{b}$ *: When maximizing the width of the block and disposing the bricks lengthwise, the minimal possible number of bricks* $= 2w$.

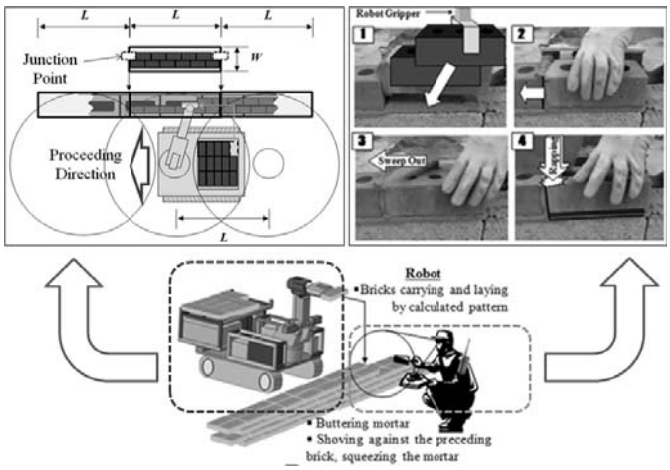


Fig.2 Brick-laying task with proposed robot system: Robot and Human

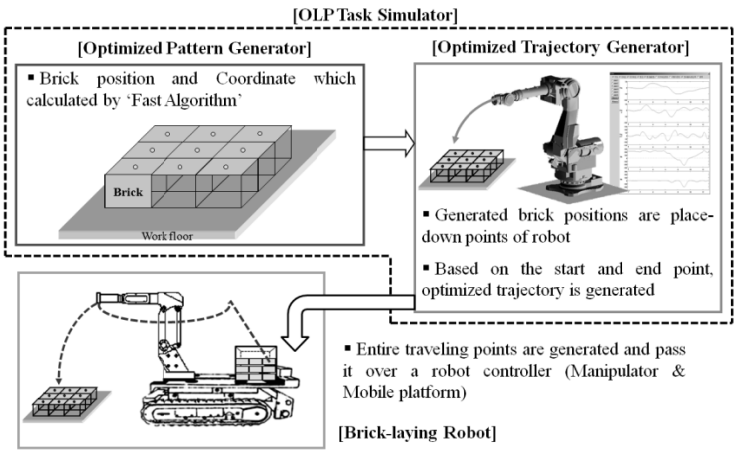


Fig. 3 Proposed system based on OLP task simulator

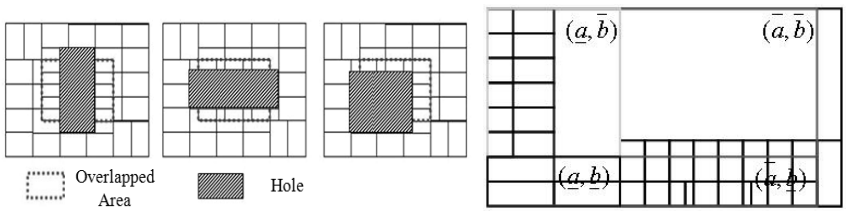


Fig.4 Treatment of the Fast algorithm and its parameter

In the first phase, (L_1, W_1), such as $(\bar{a},\bar{b})$, $(\bar{a},\underline{b})$, $(\underline{a},\bar{b})$, and $(\underline{a},\underline{b})$ are combined, and (L_1, W_1), the width and length of the other blocks can be determined.

$$(L_2,W_2)=(L_4,W_4)=\left(\left[(L-L_1)/w\right]w,\left[(W-W_1)/l\right]l\right) \quad (1)$$

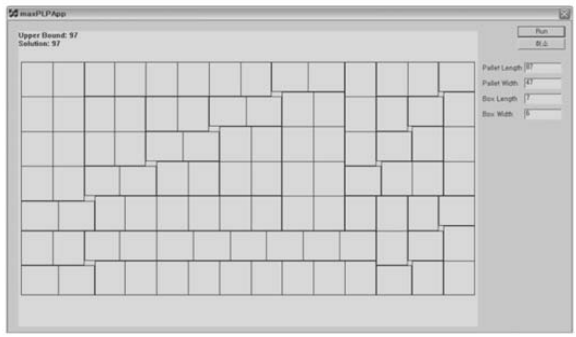
$$(L_3,W_3)=(L_1,W_1). \quad (2)$$

3.2 Computational Performance of the Fast Algorithm

The proposed algorithm is implemented on Visual C++ 6.0 and compiled with maximized-speed option. This algorithm test generated a 2D pattern of bricks and its calculation speed. As a hypothesis, the load balancing of the brick and its stability were not considered. The results of Table 2 were acquired by a computer with a K6-350-

(L, W, l, w)	Number of loaded bricks
(1000,1000,205,159)	30
(1000,1000,200,150)	33
(22,16,5,3)	23
(30,22,7,4)	23
(14,10,3,2)	23
(53,51,9,7)	42
(34,23,5,4)	38
(57,44,12,5)	41
(87,47,7,6)	97
(1200,800,176,135)	38

L	Length of laying plane
W	Width of laying plane
l	Length of brick
W	Width of brick



(a) Result of Fast Algorithm and pattern generation simulation in 2D case

(b)

(b) Brick-laying pattern generation simulation in 3D case

Fig.5 Developing the prototype brick-laying pattern generation simulator

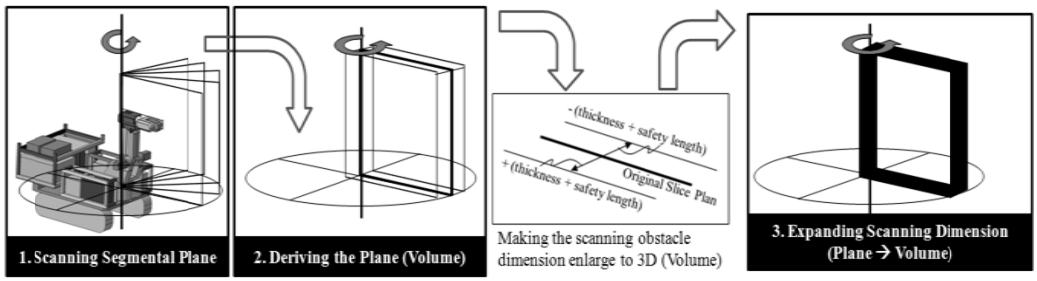


Fig.6 Expanding dimension of the segmental scanning plane

To use this algorithm practically, one dimension of height is applied additionally, and the 3D brick laying simulator is realized on like the Fig. 5 above.

4. Synthetic Trajectory Generation Algorithm

4.1 Configuration Space (C-Space) and A Algorithm: Mapping Obstacles on C-Space*

Couple of main components of the system such as palletized bricks, battery packs, and main controllers can be obstacles at the same time for the brick-laying system. Hence, the concept of C-space to solve this systematic obstacle problem was applied. The configurations defined unknown variables that exactly express the position and direction of an object, and the C-space represented all of the spaces [6]. The coordinate for each configuration was defined; each point in this coordinate that was approached to the robot gripper was expressed by the robot joint angles (configuration, posture) of the proposed system. Fig. 10 shows an example of the generation of configuration space. First, on the basis of the joint of the base frame, the imaginary plane was rotated 360 degrees. In this movement, the objects surrounding the robot were scanned, and an outline of the section was generated. The left side of Fig. 6 describes the specified brick-laying robot layout. The outline, including its interior, could be considered an obstacle. In this study, the outline was acquired by using an end effecter of the robot, and the free-movement and obstacle zones in the C-space were generated as shown in Fig. 6. To help distinguish the 3D shape of the C-space, various brightness intensities and colors are used. This figure is necessary to generate the optimal path using the A* algorithm described in the next paragraph.

4.2 Consideration of the Real Size of the Robot for Trajectory Generation

4.2.1 Modified Slice Plane

One of the disadvantages of the A* algorithm is the required computing time. The aforementioned approach considers the robot arm as a bar. Hence, the computing time load is relatively low. However, a real industrial robot has an original volume, and these factors have to be applied to the A* algorithm. The next step was to consider the real volume of the robot when it scans obstacles and generates C-obstacles. To do this, the slice planes were redefined because it was assumed that the original slice plane had no thickness, the modified slice plane had a thickness, and the factor that changed the scanning point of an obstacle of each angle was a group of both sides of the boundary of the modified slice plane (Fig. 6). The thickness of the plane was determined by the real size of the robot arm, including its gripper and load.

4.2.2 Graham's Algorithm for Convex List

Fig. 7 shows the scanning points that used the modified slice plane. The proposed system used factors of convex list point of objects and the sum of half the thickness and a safe distance. Convex list was generated by the inside apexes of objects and intersection points. If the number of intersection points was less than two, the slice plane is regarded as meeting with an apex or edge. Graham's algorithm was used to generate the convex hull which was used as the new boundary of the object when the modified slice plane was applied [7].

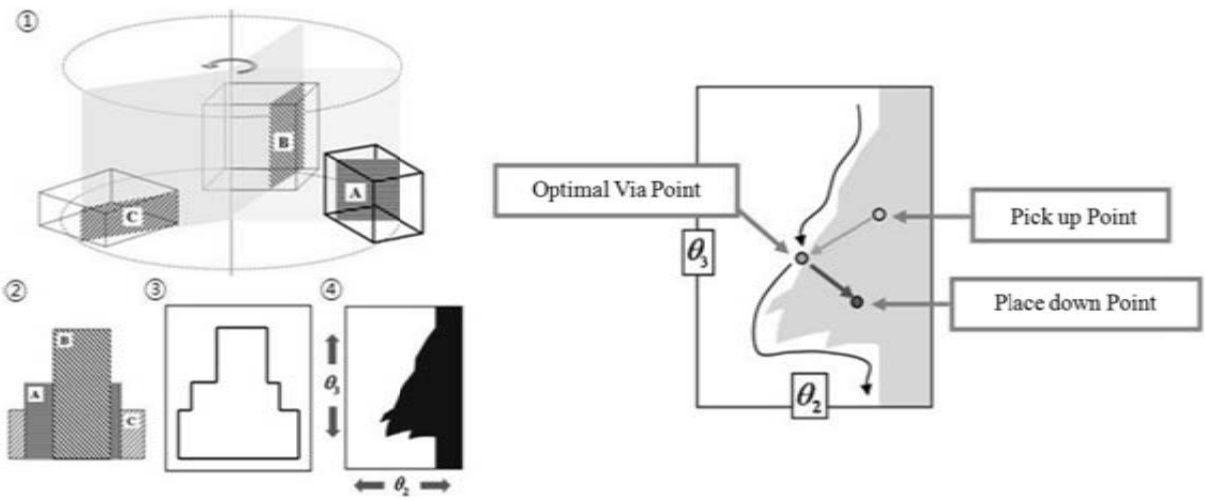


Fig.7 Procedure of Overlap Method and trajectory generation with θ_2 and θ_3

4.3 The Overlap Method for the Brick-laying Trajectory Generation

4.3.1 Basic Concept

The computing load is a critical problem in the area of software development. The main purpose of this study, as described in the Introduction, is to develop an OLP (offline programming) simulator specific to brick-laying automation. If the real size of a brick-laying robot is considered to generate the optimized trajectory, an A* algorithm is a relatively expensive method. To use this algorithm, the C-space has to be generated, but this requires a large amount of computing load. To focus on the characteristics of the brick-laying task, the new strategy which is devoted to the generation of the set of boundaries (convex) of the obstacles was proposed. As shown in Fig. 13, the proposed method overlaps the scanned images of each brick at one plane and obtains the outer line of the overlapped image. This method used the total traveling distance from the pick-up point of the bricks to the place-down point via the outer line of the overlapped area. The following equation was used to optimize the traveling distance that the robot must negotiate to deliver one brick from the pick-up to the place-down point:

$$T_{opt} = A[abs\{(P_{via} - P_{pick-up})_{\theta_2}\} + abs\{(P_{place-down} - P_{via})_{\theta_2}\} + B[abs\{(P_{via} - P_{pick-up})_{\theta_3}\} + abs\{(P_{place-down} - P_{via})_{\theta_3}\}] \quad (3)$$

4.3.2 Considerations of the Via Point

The robot path, however, is not composed of only three points (a place-down point, an optimal via point, and a place-down point). Therefore, this algorithm is exhausted to find an extra via point that would travel the whole path, from the start to the end point. To do this, the optimal via point is used as the initial point. If the gripper of the robot reaches this point, a collision between the gripper and the obstacle can be avoided by changing θ_1. The definition of the collision or gap between the robot and place-down point and the obstacle is decided beforehand.

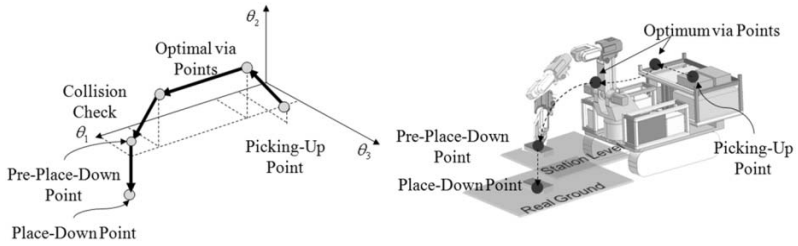


Fig.8 θ_1 for optimal via point generation

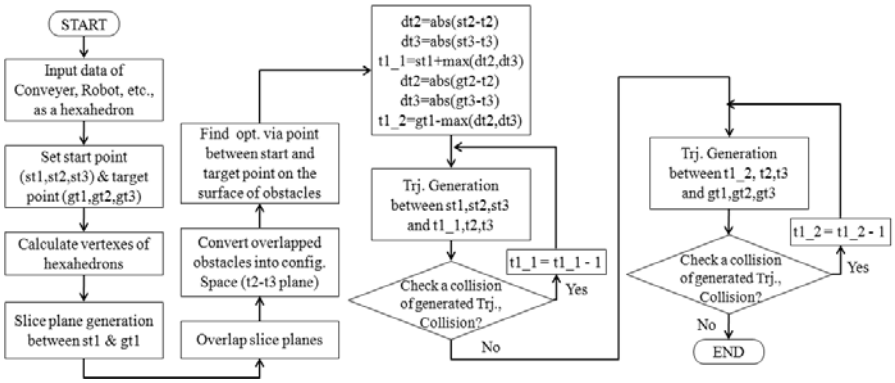


Fig.9 The overlap method algorithm

The place-down point is calculated based on the base frame plane of the manipulator. Therefore, when the robot performs the brick-paving task, the place-down point has to be considered first. Through these several treatments, the intermediate via points are decided as shown in Fig. 8.

(1) st1, st2, st3: θ_1, θ_2, θ_3 *of the starting point*

(2) gt1, gt2, gt3: θ_1, θ_2, θ_3 *of a goal point*

(3) t2, t3: θ_2, θ_3 *of an optimal path point*

(4) t1_(i): θ_1 *of an optimal path point (ith iteration)*

The total travel points are composed of [picking-up point] → 1st optimal via point, [via(θ_{11}, θ_{12}, θ_{13})] → [▪▪▪] → nth via point, [via(θ_{n1}, θ_{n2}, θ_{n3})] → final optimal via point, [via(θ_{f1}, θ_{f2}, θ_{f3})] → [place-down point]. Here, i and j of θ_{ij} mean the ith generated via point of the jth joint of the robot manipulator. Fig. 9 shows the detailed algorithm of the overlap method. This one deals with every surrounding obstacle in every unit step of the process ("unit step" means one cycle of pick-and-place task). As the shapes of the obstacles, especially palletized bricks, are changed at every step, this approach takes strong advantages on that is easy to calculate the path.

5. Result and Conclusions

To prove the efficiency of the proposed methodology, all types of trajectory generation methods described in this paper are simulated, and the results are compared. As shown in the graph, the computing time of A* algorithm that considered the volume of the robot is remarkably different depending on the situation encountered at every step (Fig. 10). However, the overlap method produces fast and stable computation results regardless of the place-down position and configurations of surrounding obstacles. Fig. 10 shows the prototype brick- laying system and simple performance tests. This is about brick positioning performance, and each position is calculated by the proposed OLP simulator based on the "Fast Algorithm" and "Overlap Method." In the next study, there will be trials for synchronizing the mobile platform with the operating algorithm to perform the brick-laying and paving task while continuously moving. Then, there will be trials for improving the dynamic performance of the mobile platform such as minimizing the hysteresis error, and keeping the constant distance precisely against the wall using by the autonomous navigation system. Lastly, there will be field-tests and make-up for a user-friendly operating interface.

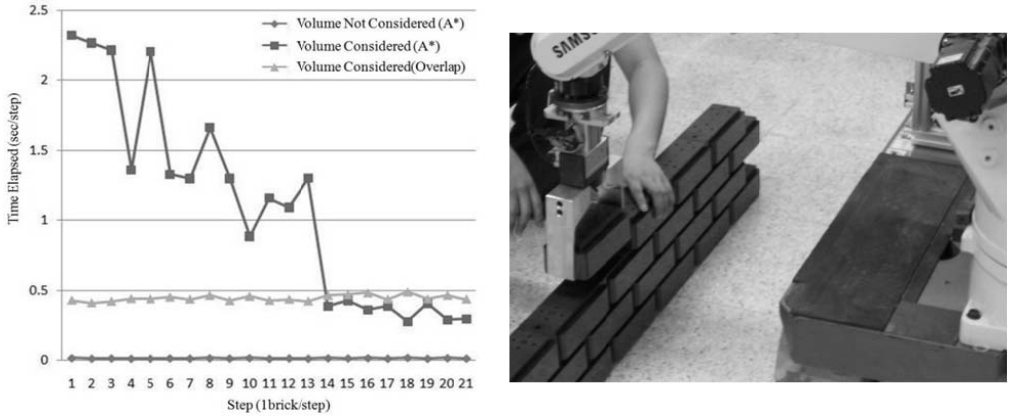


Fig.10 1-step brick-laying task and brick positioning demonstration (Brick, 21EA)

Acknowledgement

This work was supported by R&D Program of MKE(Ministry of Knowledge Economy), Research Fund of Hanyang University (HYU-2008-T) and a grant from Construction Technology Innovation Program(CTIP) funded by Ministry of Land, Transportation and Maritime Affairs (MLTM),SRC/ERC program of MOST (grant # R11-2005-056-03003-0).

References

[1] M. Miedema and P. Vink, "Metselen met de rug rechtop: ergonomische verbeteringen in de bouw [Bricklaying with the back straight: ergonomic improvements in the construction industry]," Tijdschrift voor Ergonomie, pp.38-44, "Fluitend opperen [Whistling during mechanical transport of bricks]", Uitvoering, Conference of National Society of Ergonomics, pp.154-162, 1996.

[2] Anliker, F.J. (1988). "Needs for Robots and Advanced Machines at Construction Sites", ISARC(1988), Tokyo, Japan.

[3] G. Pritschow, Dalacker, M., Kurz, J. & Gaenssle, M. (1996). "Technological aspects in the development of a mobile bricklaying robot", Automation in Construction, 5(1), 3-13.

[4] G, Y.G. & Kang, M.K. (2001). A fast algorithm for two-dimensional pallet loading problems of large size. European Journal of Operational Research, 134(1), 193-202.

[5] G. Pritschow, M. Dalacker, J. Kurz, J. Zeiher, A mobile robot for on-site construction of masonry, IEEE/RSJ/GI International Conference on Intelligent Robots and Systems (IROS '94), vol. 3, 1994, pp. 1701–1707.

[6] Warren, C.W. (1993). Fast Path Planning Using Modified A* Method. Paper presented at the IEEE International Conference on Robotics and Automation (ICRA '93), Atlanta, GA, USA

[7] Graham, R.L. (1972). An Efficient Algorithm for Determining the Convex Hull of a Finite Point Set. Info. Proc. Letters, 26, 132-133.