Noname manuscript No.
(will be inserted by the editor)

Collaboration of Multiple Autonomous Industrial Robots through Optimal Base Placements

Mahdi Hassan · Dikai Liu · Gavin Paul

Received: date / Accepted: date

Abstract Multiple autonomous industrial robots can be of great use in manufacturing applications, particularly if the environment is unstructured and custom manufacturing is required. Autonomous robots that are equipped with manipulators can collaborate to carry out manufacturing tasks such as surface preparation by means of grit-blasting, surface coating or spray painting, all of which require complete surface coverage. However, as part of the collaboration process, appropriate base placements relative to the environment and the target object need to be determined by the robots. The problem of finding appropriate base placements is further complicated when the object under consideration is large and has a complex geometric shape, and thus the robots need to operate from a number of base placements in order to obtain complete coverage of the entire object. To address this problem, an approach for Optimization of Multiple Base Placements (OMBP) for each robot is proposed in this paper. The approach aims to optimize base placements for multi-robot collaboration by taking into account task-specific objectives such as makespan, fair workload division amongst the robots, and coverage percentage, and manipulator-related objectives such as torque and manipulability measure. In addition, the constraint of robots maintaining an appropriate distance between each other and relative to the environment is taken into account. Simulated and real-world experiments are carried out to demonstrate the effectiveness of the approach and to verify that the simulated results are accurate and reliable.

M. Hassan · D.K. Liu · G. Paul
Centre for Autonomous Systems (CAS) at the University of Technology Sydney (UTS), Australia
Tel.: +612-9514-2000
Fax: +612-9514-3654
E-mail: Mahdi.Hassan@student.uts.edu.au

Keywords Autonomous Industrial Robots · Base Placement Optimization · Complete Coverage · Multi-Robot Collaboration

Mathematics Subject Classification (2000) 68T40 · 65K99

1 Introduction

The fast advancement of robotic technologies is significantly improving the processes and cost-efficiency of many manufacturing applications, and in turn enabling the shift from mass production to custom manufacturing [1]. Unlike the traditional industrial robots used for mass production where the robots are preprogrammed, and their bases are commonly fixed, Autonomous Industrial Robots (AIRs) are able to operate in complex and challenging unstructured environments. An AIR is an industrial robot, with or without a mobile platform, that has self-awareness and environmental awareness that enables it to operate autonomously in unknown or partially known environments. If the AIR is attached to a mobile platform, then its definition is the same as the Autonomous Industrial Mobile Manipulator (AIMM) [1]. AIRs are able to perform tasks such as exploration for mapping and localization [2, 3], surface-type identification [3], task or surface allocation [4], and path planning and collision-free motion planning [5, 6].

Integrating multiple cooperative AIRs can increase the capacity and flexibility of an AIR team. As an example, the two independent mobile AIRs shown in Fig. 1 are required to perform a manufacturing task (such as grit-blasting) on the three objects. For such applications, using multiple AIRs can help minimize overall completion time of the task, maximize coverage of the

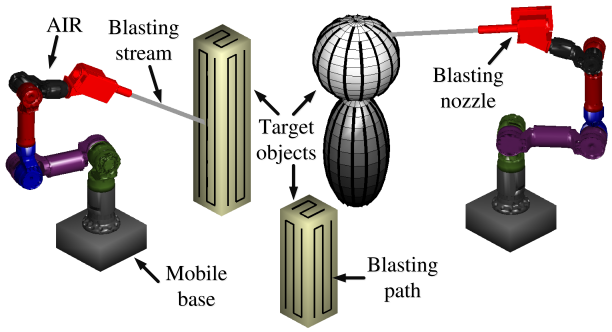


Fig. 1 Two mobile AIRs performing grit-blasting

target objects, and hence improve productivity. However, deployment of multiple AIRs requires effective collaboration amongst the AIRs during both planning and operation. The AIRs that are shown in Fig. 1 need to collaborate with each other to perform tasks such as deciding on their base placements, task allocation, and collision-free motion planning.

Surface preparation through abrasive blasting and high-pressure cleaning, surface coating and spray painting are common operations in manufacturing. *Complete coverage* is an integral part of such operations, i.e. all surface areas of interest need to be operated on. Coverage Path Planning (CPP) [7, 8, 9, 10] is the task of determining a path that passes through all points of an area or volume of interest, so as to achieve complete coverage. For example, Fig. 1 shows the grit-blasting application where the surface is cleaned by high-speed grit particles striking the surface. The stream of grit is directed by a nozzle attached to the end-effector of each AIR, which aims at a sequence of targets, i.e. a path generated using a CPP algorithm.

The work presented considers optimizing the collaboration of multiple AIRs for complete coverage tasks in manufacturing applications. The collaboration is optimized by determining which, out of a set of possible candidate base placements, should be used for each AIR in a large unstructured environment. Appropriate team objectives, which are relevant to the complete coverage task (maximal coverage and minimal makespan) and the performance of the AIRs (maximal manipulability and minimal torque), are considered.

There are research works in the available literature for finding an appropriate base placement for a single robot in different environments, such as the manufacturing environments [11, 12] and underwater environments [13, 14]. Many of the methods utilize optimization techniques given objectives that are relevant to the intended task. The problem of finding an optimal base placement for a robot can be computationally expensive due to the size of the search space and hence, researchers often simplify the problem by considering only a limited number of critical discrete end-effector positions [15] when optimizing the base placements. The objectives taken into account for the optimization are specific to the task, e.g. maximizing manipulability measure for the high accuracy needed in the milling application [16]. However, other performance measures such as the task-dependent and direction-selective performance indexes [17] are also shown to be important under certain conditions when finding an optimal base placement for a robot. The problem becomes increasingly complicated when (i) multiple robots are involved in carrying out the intended tasks [18], and (ii) when each robot must find multiple base placements such that the robot team can collectively complete the overall task.

The problem of finding a single base placement for each of the AIRs to cover a small object was investigated in [18]. This paper extends the prior work so as to enable the approach to be applicable for large objects where several base placements for each of the AIRs are needed so as to achieve complete coverage of the object under consideration. The sequence of the base placements that each AIR needs to operate from is also determined as part of the approach. In addition to the objectives relevant to the complete coverage task and the performance of the AIRs, constraints related to the robots maintaining a safe distance between each other and relative to the environment are considered.

The remainder of the paper is structured as follows. Section 2 provides a detailed description of the problem. Section 3 presents the methodology, consisting of three sub-sections: 3.1 provides an overview of the approach, 3.2 details the mathematical modeling, and 3.3 presents an approach for solving the problem where a multi-objective genetic algorithm is used. Real-world and simulated experiments are presented in Section 4 and further analyzed, compared and discussed in Section 5. Concluding comments and future studies are stated in Section 6.

2 Problem Description

Fig. 2 shows an example application where two AIRs are deployed to clean an I-beam's surfaces through grit-blasting, prior to spray painting the surfaces. At their current base placements, the AIRs can collectively cover all surfaces of the I-beam by following the boustrophedon paths, and thus achieve the complete coverage goal. However, from a different set of base placements, it may be impossible to achieve complete coverage. Thus, optimizing base placements for each AIR relative to the object and other AIRs is crucial.

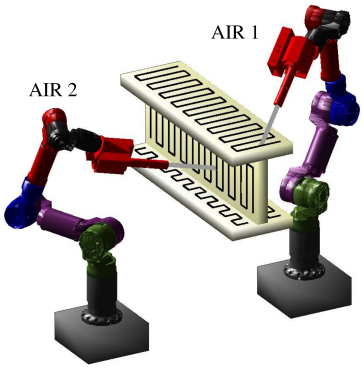
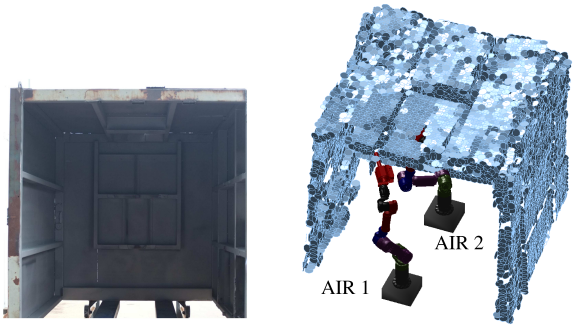


Fig. 2 The base of two AIRs positioned appropriately relative to an I-beam and each other so as to jointly achieve complete grit-blasting coverage of the I-beam



(a) A boxlike structure **(b)** Simulated scenario

Fig. 3 Two AIRs to operate on a boxlike structure and to cover all internal and external surfaces

The problem of AIR's base placement is further complicated when the object is larger than the AIR's workspace or has a complex geometric shape, and hence multiple base placements for each AIR are required to achieve complete coverage of all surfaces. For example, consider the environment shown in Fig. 3 where two AIRs are deployed to cover all internal and external surfaces of a boxlike structure. Each AIR needs to be repositioned several times so as to achieve complete coverage. Therefore, the problem is to find: (i) the minimal number, n^v of base placements for each AIR; (ii) the location of the n^v base placements for each AIR; and (iii) the visiting sequence of the n^v base placements.

The base placements problem must be solved while considering the following:

- complete (or above threshold) surface coverage;
- minimal overall completion time (or makespan);
- fair workload division between the AIRs;
- safe distance between the AIRs, and relative to the environment;
- minimal torque experienced by the AIRs' joints; and
- high manipulability of the AIRs' manipulators.

3 Methodology

In this section, first, an overview of the proposed approach to Optimization of Multiple Base Placements (OMBP) for each AIR is presented, followed by a detailed explanation of the mathematical model, and finally, the utilization of multi-objective optimization.

3.1 The OMBP Approach

The flowchart that is shown in Fig. 4 illustrates the OMBP approach and where it fits in the overall operation of the AIRs. Table 1 defines the parameters used in this paper and their notation. Sets and functions are represented as uppercase letters, scalars are represented as lowercase letters, and vectors or matrices are represented as bold lowercase letters. Superscripts are used to help describe the parameter, whereas subscripts are used as indices.

AIRs are able to perform tasks autonomously. Two of these tasks are exploration for mapping [2, 3] and localization [19], which are shown in block 1 of the flowchart. Note that appropriate control architectures [20] needs to be devised for these tasks.

The next step is communication between the AIRs for sharing of information, as shown in block 2 of the flowchart. The AIRs share the obtained information to generate a complete map of the environment. These sets of information are then used in the OMBP approach (block 3).

As shown in module 3.1 of Fig. 4, the OMBP approach starts by discretizing the search space for two main reasons: (i) the object on which the AIRs operate is large or complex, thus multiple base placements for each AIR are needed, and (ii) complete coverage requires finding feasible AIR poses for a large number of points that represent the environment or the object, which is a computationally intense process. Discretization is acceptable for many manufacturing applications if near-optimal solutions can be obtained.

Fig. 5a shows two sets of discrete base placements $B_i = \{\boldsymbol{b}_{i1}, \boldsymbol{b}_{i2}, \ldots, \boldsymbol{b}_{i(n_i^b)}\}$ for $i = 1, 2$ where i is the AIR's index and n_i^b is the total number of discrete base placements for the ith AIR. A base placement is defined as the x, y, z position with respect to a reference point. If the joint on the base of an AIR is not a full 360° revolute joint about the z-axis, then the AIR's base needs to rotate a number of times about the z-axis so as to cover

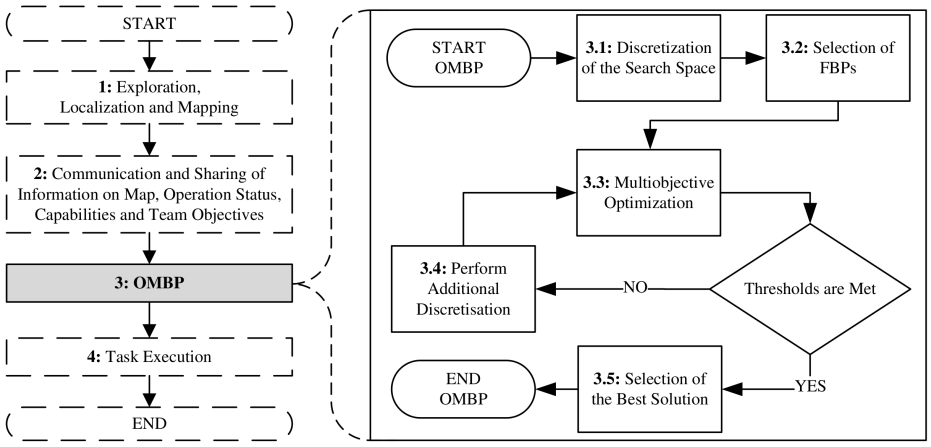
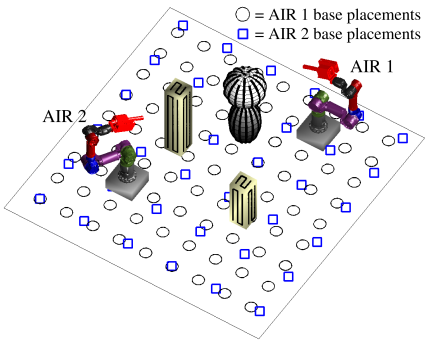
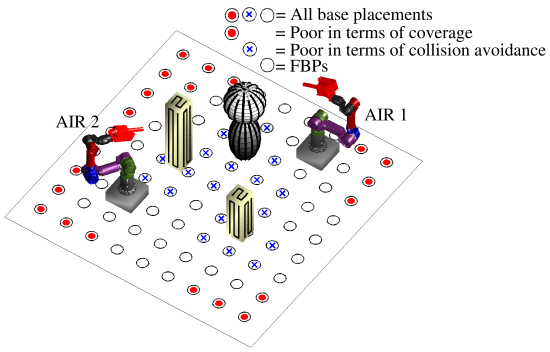


Fig. 4 A flowchart showing an overview of the OMBP approach and its modules

all the reachable areas from a base placement. For each AIR, the number of discrete base placements and their density can be decided based on the application and the capacity of the AIR such as its workspace size. For example, in Fig. 5a, the two AIRs have different capacity; thus the density of the discrete base placements is different for each AIR. If the two AIRs were identical, then the discrete base placements will be the same for both AIRs. One challenge is to select a number of base placements from the set B_i for each AIR, such that the team objectives are optimized. However, prior to performing the base placements optimization, simple preliminary filtering can prevent potentially poor performing base placements from becoming candidates, hence reducing the size of the search space. For example, Fig. 5b shows the base placements of the first AIR. The base placements with a cross are deemed to be too close to the objects and are discarded to prevent the AIR from having a high likelihood of collision with the objects. The base placements indicated with filled red circles are also prevented from becoming candidates due to their predicted low coverage of the objects. The rest of the base placements are anticipated to have reasonable to high coverage and are an acceptable distance away from the objects. These base placements are henceforth referred to as Favored Base Placements (FBPs). This process of finding the FBPs is represented in module 3.2 of the flowchart.



(a) Discrete base placements of two AIRs



(b) FBPs of AIR 1

Fig. 5 Discrete base placements of two AIRs with different capacity, and the FBPs of the first AIR

After determining the FBPs of each AIR, the next step is to perform the multi-objective optimization (module 3.3). The aim is to select a subset of FBPs for each AIR and to determine the visiting sequence of the selected FBPs such that the team objectives are optimized, and constraints are satisfied.

The output of the multi-objective optimization is the Pareto optimal solutions that lie on the Pareto front. The Pareto front will be discussed in Section 3.3.

Coarse discretization of the search space can be initially considered; however, if certain thresholds are not met (e.g. if the overall coverage is not above a threshold) then a finer discretization (module 3.4) can be generated around the good performing base placements. In doing so, a time-efficient option is to keep the best so-

Table 1: Nomenclature

Parameter	Definition
B_i	A set of discrete base placements for the ith AIR
B_i^{FBP}	A subset of base placements from the set B_i, called Favored Base Placements (FPBs)
$\boldsymbol{b}_{ij}$	jth discrete base placement from the set B_i
$\boldsymbol{\beta}_i$	A favored base placement from the set B_i^{FBP}, associated with the ith AIR
$\boldsymbol{\beta}_i^{AIR}(t)$	Base placement of the ith AIR at time t
d_i	Distance between two adjacent targets along a path of the ith AIR
$F_j(Z)$	A function that calculates the value of the jth objective based on the design variables in Z
δ	Minimum distance threshold between the base placements of two AIRs
δ_{ik}^s	A small negative or positive integer to be added to g_{ik}
g_{ik}	kth nonzero gene in the ith part of a chromosome
I^s	A set containing the indices of the progress times in T^s
$\mathbf{J}(\boldsymbol{q}_{ikj}^f)$	A function that calculates the Jacobian of the pose $\boldsymbol{q}_{ikj}^f$
$N^f(Z_{ik})$	A function that calculates the number of targets that can be reached with feasible poses of the ith AIR at the kth base placement
n	Number of AIRs deployed
n_i^b	Number of discrete base placements in the set B_i
n_i^D	Number of nonzero genes selected from dad's chromosome for the ith part
n_i^F	Number of favored base placements (i.e. size of the set B_i^{FBP})
n_i^g	Number of genes in the ith part of a chromosome (i.e. the length) corresponding to the ith AIR
n_i^M	Number of nonzero genes selected from mom's chromosome for the ith part
n^j	Number of joints of an AIR
n_i^O	Number of targets associated with the ith AIR
n_i^T	Total number of targets associated with the ith AIR which represent all surfaces
n_i^v	Number of base placements to be visited by the ith AIR
n^v	Number of base placements to be visited by all AIRs

Parameter	Definition
O_i	A set of discrete points (called *targets*) that are associated with the ith AIR and are used to represent all surfaces
O_i^{al}	A set containing the targets that are allocated to the ith AIR
O_{ik}	A set of targets that are inside the workspace of the ith AIR at the kth base placement
$\boldsymbol{o}_{ikj}$	jth target in the set O_{ik}
$\boldsymbol{q}_{ikj}^f$	A feasible AIR pose that reaches the target $\boldsymbol{o}_{ikj}$ with correct end-effector position and orientation, and without collision
$\mathrm{T}_i(Z)$	A function that calculates the overall completion time of the ith AIR based on relevant design variables in Z
T^s	A set containing the progress times of the n AIRs sorted from the lowest time to the highest
$\mathcal{T}_i^{al}$	A set containing the maximum torque ratio corresponding to each target in O_i^{al}
$\mathcal{T}_{im}(\boldsymbol{q}_{ikj}^f)$	A function that calculates the torque experienced by joint m of the ith AIR at pose $\boldsymbol{q}_{ikj}^f$
$\mathcal{T}^{Rmax}(\boldsymbol{q}_{ikj}^f)$	A function that calculates the maximum torque ratio of the pose $\boldsymbol{q}_{ikj}^f$
t_i	Current progress time of the ith AIR
t^c	Overall completion time of the task (makespan)
t_i^s	Time associated with the ith AIR setting-up and moving to the next base placement
τ_{im}^{cap}	Torque capacity of joint m of the ith AIR
v_i	End-effector speed of the ith AIR
v_i^d	Difference between the maximum and minimum End-effector speeds of the ith AIR
v_i^{min}	Minimum end-effector speed of the ith AIR
v_i^{max}	Maximum end-effector speed of the ith AIR
$W(\boldsymbol{q}_{ikj}^f)$	A function that calculates the manipulability measure of the pose $\boldsymbol{q}_{ikj}^f$
W_i^{al}	A set containing the manipulability measure corresponding to each target in O_i^{al}
w_{ikj}	A weighting factor (from 0 to 1) applied to the end-effector speed of the ith AIR based on the area in which the target $\boldsymbol{o}_{ikj}$ is located
Z	A set containing all the nonzero design variables
Z_{ik}	kth design variable associated with the ith AIR
θ_j	Angle of the jth joint of an AIR pose

lutions obtained from the previous optimization run to use as part of the initial solution/s for the next optimization run. For example, if Genetic Algorithm is used, then the initial population can be made up of the best solutions from the previous run combined with the new base placements generated from finer discretization.

Using the OMBP approach, the index and the visiting sequence of the FBPs that each AIR needs to visit is determined, which is followed by the task execution (block 4). During the task execution and at each assigned base placement of the AIRs, collision-free motion planning [5] is to be performed by the AIRs in order to cover the paths generated on the surfaces of the object to obtain complete coverage [7]. Prior to the motion planning at each base placement, area partitioning and allocation [4] may also be required to appropriately partition the overlapped areas that fall inside the overlapped workspace of the AIRs and to allocate the partitioned areas amongst the AIRs equitably.

3.2 Mathematical Modeling

In this section, firstly the design variables are presented followed by the design objectives. An explanation of how the objectives conflict with each other is also provided after introducing each objective. Finally, the design constraints are expressed.

3.2.1 Design Variables

Let $B_i^{FBP} = \{\boldsymbol{\beta}_{i1}, \boldsymbol{\beta}_{i2}, \ldots, \boldsymbol{\beta}_{i(n_i^F)}\} \subseteq B_i$ be the FBPs associated with the ith AIR, for $i = 1, 2, \ldots, n$. Note that for the ith AIR, $n_i^F \leq n_i^b$, meaning that the number of FBPs are less than or equal to the total number of discrete base placements. The design variables are $Z_{ik} \in \{0, 1, \ldots, n_i^F\}$ with constraints $Z_{ij} \neq Z_{ik} \iff Z_{ik} > 0$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n_i^F$, $k = 1, 2, \ldots, n_i^F$, $k \neq j$. As an example, the design variable $Z_{ik} = 3$ means that the kth base placement of the ith AIR is the third FBP, i.e. the ith AIR is to visit $\boldsymbol{\beta}_{i(Z_{ik})} = \boldsymbol{\beta}_{i3}$ for its kth base placement. Hence, the ith AIR can visit

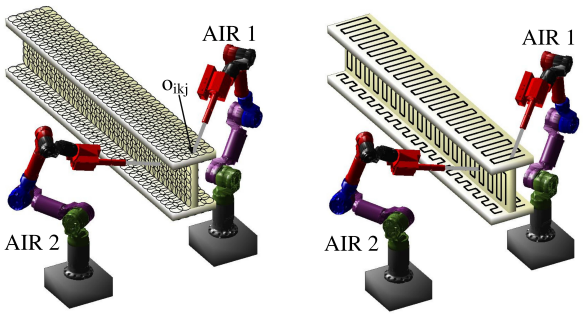
up to a maximum of n_i^F FBPs where n_i^F is the total number of FBPs. If a design variable is given a value of zero, i.e. if $Z_{ik} = 0$, then one less FBP will be visited by the ith AIR and the AIR will move from the $(k-1)$th base placement $\boldsymbol{\beta}_{i(Z_{ik-1})}$ to the $(k+1)$th base placement $\boldsymbol{\beta}_{i(Z_{ik+1})}$. Let Z be a set containing all the design variables that have a value greater than 0, i.e. $Z = \{Z_{ik} | Z_{ik} > 0, \forall i, k : i = 1, \ldots, n, k = 1, \ldots, n_i^F\}$. The AIRs collectively visit n^v FBPs where n^v equals the number of design variables in Z. Similarly, n_i^v is defined as the number of FBPs to be visited by the ith AIR only, and can be determined based on the number of nonzero design variables that are associated with the ith AIR. Ultimately, the aim is to obtain values for n^v nonzero design variables such that the team objectives are optimized while constraints are satisfied.

It needs to be noted that since the number of FBPs to be visited by an AIR is initially unknown, the extreme, but unlikely case would necessitate visiting all FBPs, and hence the number of design variables can be as large as the number of FBPs. However, an approximation of the number of FBPs to be visited by each AIR, i.e. n_i^v, can be made based on the size of the object (explained in Section 3.3). This approximation can significantly reduce the size of the search space.

3.2.2 Design Objectives

The main objectives that are relevant to the task of complete surface coverage and the performance of the AIRs are (i) maximal coverage, (ii) minimal makespan, (iii) maximal manipulability measure, and (iv) minimal AIRs' joint torques.

Objective 1 - *Maximal Coverage:* It is vital that the base placements selected by the AIRs result in maximum coverage of the surfaces. Fig. 6 shows two AIRs that are deployed to perform the task of grit-blasting. At each assigned base placement of an AIR, a set of discrete points, $O_{ik} = \{o_{ik1}, o_{ik2}, \ldots, o_{ik(n_i^O)}\}$, which are used to represent the surfaces of an object, are located inside the workspace boundary of the AIR where k is the base placement index of the ith AIR and n_i^O is the total number of discrete points that falls inside the workspace boundary of the ith AIR. These discrete points will henceforth be referred to as *targets*. Due to constraints such as the joint angle limits of the AIR, some of these targets may be unreachable by the AIR. In order for a target, $\boldsymbol{o}_{ikj} \in O_{ik}$ to be *reachable* by the ith AIR, a feasible AIR pose $\boldsymbol{q}_{ikj}^f = [\theta_1, \theta_2, \ldots, \theta_{n^j}]$ needs to be found for the target where θ_1 to θ_{n^j} are the angles of the n^j AIR joints.



(a) An I-beam represented using targets

(b) Paths generated by joining targets' centroid

Fig. 6 Two AIRs to cover all surfaces of an I-beam

A feasible AIR pose is one that can reach the target with appropriate end-effector orientation and position, and without collision, e.g. the pose of AIR 1 ($i = 1$) shown in Fig. 6a generated to cover the target $\boldsymbol{o}_{ikj}$. One option for computing a feasible AIR pose for a target is to use the lookup table explained in [18]. Another option is to perform inverse kinematics using a numeric approach (e.g. optimization based) or an analytical approach (if possible) and then performing collision checking to assess feasibility. Determining a feasible AIR pose should account for minimal torque on the AIR's joints and maximal manipulability measure, for example as explained in [18], so as to calculate the following objectives more accurately.

At each assigned base placement of an AIR, the reachable targets can be joined to form a path using a single robot coverage path planning algorithm [7]. Ideally, the generated paths of the AIRs at all the selected base placements result in complete coverage paths, such as the paths shown in Fig. 6b, which cover all surfaces of the object. Thus, to achieve complete surface coverage, the base placements of all AIRs should be selected such that all targets representing the surfaces can be reachable by feasible poses of the AIRs. The size of the targets shown in Fig. 6a are the same for both AIRs, since in this example both AIRs are identical. However, depending on the capacity of each AIR, the size and the density of the targets can be different, meaning that the paths generated on the surfaces of an object can also be different for each AIR.

This objective is therefore to maximize the number of targets that can be reached by all AIRs. That is, to minimize missed-coverage:

$$\min_{Z} F_1(Z) = 1 - \sum_{i=1}^{n} \frac{\sum_{k=1}^{n_i^v} N^f(Z_{ik})}{n_i^T} \tag{1}$$

where Z is a set containing all the design variables that have a value greater than zero, n is the number of AIRs, n_i^v is the total number of base placements that the ith AIR needs to visit, n_i^T is the total number of targets associated with the ith AIR which represent all surfaces, and $N^f(Z_{ik})$ calculates the number of targets that can be reached with feasible AIR poses at a base placement decided based on Z_{ik} which is a design variable in Z associated with the ith AIR and its value determines the FBP to be visited by the ith AIR for the kth base placement. Since the AIRs can have different capacities, then each AIR can be associated with a different set of targets that represent the surfaces. Hence, n_i^T can be different for each AIR (i.e. for each i). The overlapped targets, which more than one AIR can reach, need to be counted only once. The overlapped targets can be found by performing a simple distance query, and can be partitioned and allocated based on the method in [4] or based on the "first come, first served" basis. Note that in Eq. (1), $F_1(Z) = [0, 1]$ and represents the percentage of missed-coverage. A value of 0 for $F_1(Z)$ corresponds to an optimal result, meaning that there is no missed-coverage and all areas of interest are covered. Conversely, a value of 1 corresponds to the worst possible result, meaning that the AIRs could not cover any section of the surface.

Objective 2 - *Minimal Makespan:* The second objective is to minimize the makespan (i.e. the overall completion time of the task). Optimizing this objective has the added benefit of equitably dividing the workload between the AIRs, since, in order to achieve the minimal makespan, the coverage task needs to be divided appropriately amongst the AIRs. This objective also takes into account the set-up time associated with repositioning an AIR. If the cost of repositioning is set appropriately, then minimizing the makespan can also minimize the number of base placements needed. Thus, this objective is to minimize the makespan:

$$\min_{Z} F_2(Z) = \max\{\mathrm{T}_1(Z), \mathrm{T}_2(Z), \ldots, \mathrm{T}_n(Z)\} \tag{2}$$

where $\mathrm{T}_i(Z)$ is the completion time of the ith AIR, which can be calculated as

$$\mathrm{T}_i(Z) = \left(\sum_{k=1}^{n_i^v} N^f(Z_{ik}) \cdot \frac{d_i}{v_i} \right) + n_i^v \cdot t_i^s \tag{3}$$

where n_i^v is the total number of base placements that the ith AIR needs to visit, $N^f(Z_{ik})$ calculates the number of targets that can be reached with feasible AIR poses at a base placement decided based on the value of Z_{ik} in Z, d_i is the distance between two adjacent targets along a path of the ith AIR, v_i is the chosen end-effector speed of the ith AIR suitable for the application, and t_i^s is the set-up time associated with the ith AIR moving to the next base placement, which may include tasks such as turning off/on accessories and tools.

Note that this objective (minimal makespan) is in conflict with the Objective 1 (maximal coverage in Eq. (1)). The conflict is due to the following reasons: (i) the aim of Objective 1 is to minimize missed-coverage by increasing the number of targets that each AIR covers which in turn increases the makespan, hence Objective 1 is optimal when all areas are covered whereas Objective 2 is optimal when no area is covered, and (ii) Objective 1 has the potential to cause a larger number of FBPs to be selected for each AIR so as to maximize coverage by reaching more targets, whereas Objective 2 aims to minimize the number of FBPs to be selected for each AIR since the second term in Eq. (3) (after the plus sign) considers a penalty to account for the set-up time. Due to the conflict in these objectives and the following objectives, a multi-objective optimization algorithm is appropriate to solve the mathematical model.

Equation 3 assumes that a constant end-effector speed will be employed to achieve uniform coverage of the surfaces (e.g. when spray painting a vehicle). However, in certain situations and applications, non-uniform coverage may in fact be required. As an example consider the task of grit-blasting a rusted object where only certain areas of the object are heavily rusted. The rusted areas will need a more intensive, or extended period of grit-blasting to achieve a uniform surface finish. This area-specific focus can be accomplished by reducing the end-effector speed for such target areas. Each target can thus be weighted based on the surface area it falls in (e.g. the level of rust). This weighting can then be used for calculating $\mathrm{T}_i(Z)$. That is,

$$\mathrm{T}_i(Z) = \left(\sum_{k=1}^{n_i^v} \sum_{j=1}^{N^f(Z_{ik})} \frac{d_i}{(w_{ikj} \cdot v_i^d) + v_i^{min}} \right) + n_i^v \cdot t_i^s \tag{4}$$

where $v_i^d = v_i^{max} - v_i^{min}$, and where v_i^{max} and v_i^{min} are the maximum and minimum end-effector speeds of the ith AIR, respectively, and w_{ikj} is the weighting factor, $0 \leq w_{ikj} \leq 1$, applied to the end-effector speed based on the area in which the target $\boldsymbol{o}_{ikj}$ is located.

Objective 3 - *Maximal Manipulability Measure:* Performance metrics [21] such as the manipulability measure, the dexterity index, the minimum singular value, and measures of isotropy can be used to help obtain a measure for a manipulator or a manipulator pose corresponding to a certain point in the workspace. The use, limitations, and benefits of each of these measures

depend on the application and the structure of the system or the robot manipulator. Manipulability measure [22] can be used to obtain a measure for a manipulator pose corresponding to a target in the environment. The aim is to position the AIRs such that the targets are reached with poses that will increase the likelihood of finding a feasible trajectory during the task execution. The higher the sum of the manipulability measures of the poses corresponding to the targets, the higher the likelihood of finding a feasible trajectory during the task execution, since a large manipulability measure of a pose corresponds to a pose that is far away from singularities and can move more freely. Therefore, this objective is to maximize the sum of manipulability measures for all AIR poses corresponding to all targets representing the environment. That is,

$$\max_{Z} F_3(Z) = \sum_{i=1}^{n} \sum_{k=1}^{n_i^v} \sum_{j=1}^{N^f(Z_{ik})} W(\boldsymbol{q}_{ikj}^f) \tag{5}$$

where

$$W(\boldsymbol{q}_{ikj}^f) = \sqrt{\det\left(\mathbf{J}(\boldsymbol{q}_{ikj}^f)\mathbf{J}^{\mathsf{T}}(\boldsymbol{q}_{ikj}^f)\right)} \tag{6}$$

is the manipulability measure (a value from 0 to 1), n_i^v is the total number of base placements that the ith AIR needs to visit, $N^f(Z_{ik})$ calculates the number of targets that can be reached with feasible AIR poses at a base placement decided based on the value of design variable Z_{ik} in Z, $\mathbf{J}(\boldsymbol{q}_{ikj}^f)$ is the Jacobian of the pose $\boldsymbol{q}_{ikj}^f$. When determining a feasible pose for an AIR, maximizing manipulability measure for the AIR needs to be considered, e.g. as per the lookup table in [18].

This objective (maximal manipulability) is in conflict with Objective 2 (minimal makespan). Objective 2 aims to minimize the makespan which indirectly minimizes the number of targets to be covered and the number of FBPs for each AIR to operate from. On the contrary, this objective has the potential to achieve a greater sum of manipulability measures, $F_3(Z)$ (in Eq. (5)), when more targets are covered and possibly when the AIRs operate from a larger number of FBPs.

This objective (maximal manipulability) and Objective 1 (maximal coverage) benefit from covering a greater number of targets. However, greater coverage does not necessarily equate to a greater sum of manipulability measures. Hence, this objective is required. For example, there may be a number of base placements from which an AIR can cover the same targets; however, from one of these base placements the AIR may be able to reach the targets with better manipulability measure. That is, there can be multiple solutions for which coverage is maximal, but not all solutions are the same in terms of manipulability measure, and vice versa. Objective 3 and Objective 1 may be combined using the weighted sum method which requires proper normalization of the objectives. In this case, Objective 1 should be given a greater weight since maximizing coverage is more important in complete coverage tasks. However, due to the suitability of multi-objective optimization with respect to other objectives, it is preferable for Objectives 1 and 3 not to be combined to enable straightforward selection from, and comparison between, the Pareto optimal solutions.

Objective 4 - *Minimal Torque:* To improve the operating condition of an AIR, it is preferable to minimize the torque experienced by the joints of the AIR.

Let the torque ratio of joint m of a feasible AIR pose $\boldsymbol{q}_{ikj}^f$ be the amount of torque the joint has experienced divided by its torque capacity. The maximum torque ratio $\mathcal{T}^{Rmax}(\boldsymbol{q}_{ikj}^f)$ for the pose $\boldsymbol{q}_{ikj}^f$ is the largest torque ratio from all the joints of the ith AIR, i.e.

$$\mathcal{T}^{Rmax}(\boldsymbol{q}_{ikj}^f) = \max_{m} \left| \frac{\mathcal{T}_{im}(\boldsymbol{q}_{ikj}^f)}{\tau_{im}^{cap}} \right| \tag{7}$$

where $\mathcal{T}_{im}(\boldsymbol{q}_{ikj}^f)$ is the torque experienced by joint m of the ith AIR at pose $\boldsymbol{q}_{ikj}^f$, and τ_{im}^{cap} is the toque capacity of the mth joint.

This objective is to minimize the sum of maximum torque ratios of the AIR that experiences the most amount of torque. That is,

$$\min_{Z} F_4(Z) = \max_{i} \left(\sum_{k=1}^{n_i^v} \sum_{j=1}^{N^f(Z_{ik})} \mathcal{T}^{Rmax}(\boldsymbol{q}_{ikj}^f) \right) \tag{8}$$

where n_i^v is the total number of base placements that the ith AIR needs to visit, and $N^f(Z_{ik})$ calculates the number of targets that can be reached with feasible AIR poses based on the design variable Z_{ik} in Z.

For a feasible AIR pose $\boldsymbol{q}_{ikj}^f$, the torque at each joint is calculated based on: (i) the weight of the nozzle, joints (actuators), and links of the AIR; and (ii) the reaction force generated on the nozzle, e.g. due to the stream of grit or paint exiting the nozzle. Readers are advised to refer to [23] for information on calculating torque for an AIR manipulator pose. When finding a feasible pose for an AIR at a base placement, the feasible AIR pose needs to be determined such that the torque on AIR's joints is minimized, e.g. using the lookup table in [18]. Note that in the applications under consideration, such as grit-blasting and spray painting, the AIR moves at a slow speed when operating on a surface. Hence, torque due to angular, centripetal, and Coriollis accelerations [23] can be neglected.

This objective (minimal torque) is in conflict with Objective 1 (maximal coverage) and Objective 3 (maximal manipulability). Both Objectives 1 and 3 benefit from covering a larger number of targets, whereas similar to Objective 2 (minimal makespan), this objective (minimal torque) is negatively affected by larger coverage since the sum of torque experienced by the AIRs increases as coverage increases (not at a constant rate). It may seem then that this objective can be combined with Objective 2 (minimal makespan) since they both benefit from lower coverage. However, this objective can in fact be in conflict with Objective 2. In Objective 4, selecting a larger number of base placements from which the AIRs can operate, may result in more reachable targets and reduced torque experienced by the AIRs, whereas in Objective 2 there is a set-up time penalty proportional to the selected number of base placements so as to minimize makespan.

3.2.3 Design Constraints

Constraint 1 *- Distance Between Any Two AIRs:* The proximity of the AIRs with respect to each other at any time during the task execution should not be allowed to cause restrictions on their maneuverability or cause a high risk of collision. A minimum acceptable distance between the AIRs or a threshold δ should be determined based on the application or the structure of the AIRs. For example, for the AIRs shown in Fig. 6, δ can be the distance from an AIR's base to the workspace boundary of the AIR. For simplicity, the boundary can be approximated to be a sphere. Thus, the AIRs' base placement should be chosen such that

$$\left\|\boldsymbol{\beta}_i^{AIR}(t) - \boldsymbol{\beta}_l^{AIR}(t)\right\| > \delta \tag{9}$$

$\forall i, l : i = 1, \ldots, n, l = 1, \ldots, n, i \neq l$, and $\forall t : t = 0, \ldots, t^c$ where $\boldsymbol{\beta}_i^{AIR}(t)$ and $\boldsymbol{\beta}_l^{AIR}(t)$ are the base placements of the ith and lth AIRs at time t, respectively, n is the total number of AIRs, and t^c is the overall completion time of the task. An alternative option is to design this constraint as an objective function that maximizes the distance between the AIRs. Although this option may be helpful for some applications, it does not guarantee that it will satisfy the constraint of maintaining a safe distance between the AIRs. Thus, the solution needs to be checked afterwards for feasibility.

Constraint 2 *- Distance to Obstacles:* Note that the constraint on the distance between any AIR and the environment or the objects (obstacles) is already considered as part of the selection of FBPs. Recall that during the selection of FBPs, the base placements that are in close proximity to the objects are discarded.

3.3 Multi-objective Optimization

An appropriate optimization algorithm needs to be utilized to solve and test the proposed mathematical model. Multi-objective optimization algorithms [24] are useful for simultaneously optimizing multiple objectives that can be in conflict with each other. The output of the multi-objective optimization algorithms is the Pareto front. An advantage of obtaining the Pareto front is that the strategy for selecting the final solution from the Pareto front can be conveniently modified to suit the changes that occur within the application without the need for repeating the optimization.

For the last two decades, strong and continuing research has been dedicated to the development of evolutionary algorithms [25]. An example of Multi-Objective Evolutionary Algorithms (MOEAs) is Non-dominated Sorting Genetic Algorithm II (NSGA-II) [26], which can be a suitable option for solving the proposed mathematical model.

Metaheuristic algorithms such as NSGA-II can be helpful in addressing NP-hard problems, particularly for problems with high combinatorial complexity and discrete search space. The optimization problem under consideration is a combination of the well-known Art Gallery Problem (AGP) and the Multiple Traveling Salesmen Problem (MTSP) [27, 28], both of which are considered to be NP-hard. AGP asks for the minimum number of points (and their locations) from which the entire environment can be observed, which is, in the problem under consideration, the same as finding the minimum number of FBPs (and their locations) from which the environment can be covered. In the MTSP, the goal is to find the visiting sequence of a number of cities by multiple salesmen such that the total cost (e.g. travel time) is minimized and constraints are satisfied. This is similar to finding the visiting sequence of the selected FBPs for each AIR such that AIR team's objectives and constraints are met. Thus, NSGA-II is suitable for the problem under consideration, and its effectiveness is verified using experimental results presented later in this paper. Note that comparing NSGA-II to other metaheuristic optimization algorithms that can solve the proposed mathematical model is outside the scope of this paper.

3.3.1 Chromosome Representation

A chromosome representation is developed that is designed and tested specifically for the problem under consideration. The work in [27] uses a two-part chromosome representation to solve the Multiple Traveling Salesmen Problem (MTSP), and the authors explain

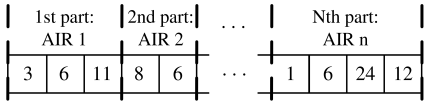


Fig. 7 A multi-part chromosome representation developed for the problem under consideration

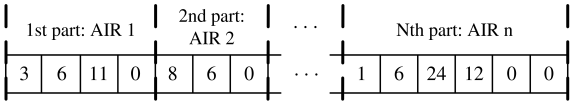


Fig. 8 Multi-part chromosome representation with additional zero genes to deal with the uncertainty in determining the number of base placements to visit

that the two-part chromosome reduces the search space when compared to one-chromosome and two-chromosome representations. The developed chromosome representation is to a certain extent similar to the two-part chromosome [27]; however, there are two main differences. As shown in Fig. 7, the first difference is that instead of having two parts for each chromosome, multiple parts are considered where each part is associated with one of the AIRs. The second difference is that instead of having a part in the chromosome to represent the required number of base placements for each AIR, i.e. for determining the value of $n_i^v, \forall i : i = 1, \ldots, n$, this parameter is made to correspond to the fixed length of the ith part of the chromosome associated with the ith AIR.

Fig. 7 shows an example of the developed chromosome representation where the first part of the chromosome corresponding to the first AIR has a length of three. That is, there are three nonzero genes in this part, which represent the indices of the FBPs that the first AIR needs to visit. The visiting sequence of the selected FBPs is from left to right. Therefore, in this example, AIR 1 first visits the 3rd FBP followed by the 6th FBP and then the 11th FBP.

Alternatively, each part of the chromosome can be made with a length equaling the number of FBPs of the corresponding AIR, and binary encoding can then be used to determine the FBPs that need to be visited by the AIR. However, to reduce the length of the chromosome and also to reduce the search space, each part is generated with a length based on the capacity of the corresponding AIR. For example, for two identical AIRs with the same capacity, if either AIR can individually cover the entire object using a minimum of 8 FBPs, then the length of each part corresponding to each of the two AIRs is 4 (i.e. $8/2 = 4$). Meaning that if 8 appropriate FBPs are chosen for both AIRs (4 for each AIR), then the entire object is covered. If the AIRs' capacities are different, then the length of each part of the chromosome can take into account the capacity of the AIRs. In Fig. 7, the length of the first, second and nth part of the chromosome is 3, 2 and 4, respectively, meaning that AIR 2 has a greater capacity (e.g. larger workspace size) than AIR 1, and AIR 1 has greater capacity than AIR n. Besides reducing the search space and chromosomes' length, another advantage of this chromosome representation is that when additional base placements are considered for finer discretization of the search space (Module 3.4 of Fig. 4), the length of the chromosome doesn't need to be changed. This is because the number of base placements to be visited by each AIR will remain the same.

It may not be possible to accurately determine the number of base placements n^v needed for all AIRs to collectively cover the entire object. Thus, a reasonable approximation can be used based on the size of the object, or the number and the density of the targets that represent the object. The initial population can be generated such that for each part of a chromosome there are additional genes with a value of zero, as shown in Fig. 8. When creating a new population at each GA iteration, chromosomes with a different number of zero genes can be made, e.g. through crossover operation. The genes with the value of zero are interpreted as void (i.e. they don't represent any base placement). Thus, incorporating additional genes provides the flexibility to increase or reduce the number of FBPs to be visited by each AIR. The greater the uncertainty in approximating n^v for an application or an environment, then the larger the number of zero genes that can be added to the chromosome when generating the initial population. Note that the requirements that will be outlined in the following explanation of the crossover and mutation operators need to be accounted for so as to prevent infeasible solutions being added to the initial population and subsequent population.

3.3.2 Crossover Operator

An example crossover operator is presented in this section, and the requirements that need to be taken into account for achieving feasible solutions are explained. An offspring generated from a crossover operator is only added to the subsequent population if it satisfies all the relevant requirements; otherwise, it is discarded. Alternatively, the crossover operator can be implemented such that an offspring automatically meets the requirements as part of the way the offspring is generated.

In performing a crossover operation, a pair of parent chromosomes are selected, and a child chromosome is generated from the parents in the hope that the

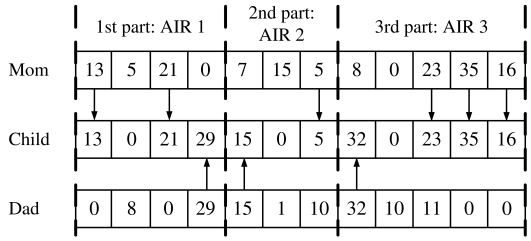


Fig. 9 An example of the crossover operation for the developed multi-part chromosome representation

child chromosome will provide a better solution. There are several methods for selecting and exchanging genes from the parents chromosomes to form the child chromosome. Uniform crossover [29, 30] with some modifications is used for the developed chromosome representations; however, other crossover methods can also be used. Fig. 9 is an example where crossover operation is performed on two selected parents to generate the child chromosome. There are three parts for each chromosome meaning that there are three AIRs. The length of each part of the chromosome is different since each AIR has a different capacity. It can be seen that for each part of the chromosome a random number of nonzero genes are first selected from the dad's chromosome, then a random number of genes are selected from the mom's chromosome, and finally the rest of the genes are given a value of zero.

Several requirements need to be taken into account so as to obtain feasible solutions when designing the crossover operator for the problem under consideration:

1. Let n_i^g be the number of genes in the ith part of a chromosome (i.e. the length) corresponding to the ith AIR. n_i^v nonzero genes are to be selected from the parents chromosomes for the ith part of a child chromosome such that $n_i^v \leq n_i^g$. n_i^v can be based on the n_i^v value of one of the parents chromosomes or can be a randomly generated value with the lower bound being the minimum number of base placements for the ith AIR. The remaining number of genes, i.e. $n_i^g - n_i^v$ genes, are assigned zeros.
2. n_i^D nonzero genes are selected from dad's chromosome and n_i^M nonzero genes are selected from mom's chromosome for the ith part of a child chromosome such that $n_i^D + n_i^M = n_i^v$. Selection of the genes from the dad's and mom's chromosomes can be random, left to right, etc. The selected nonzero genes can then be copied on the same genes of the child's chromosome, copied from left to right, or randomly.
3. Let g_{ik} be the kth nonzero gene in the ith part of a chromosome. $g_{ik} \neq g_{im}$, i.e. in the ith part of a chromosome, the kth nonzero gene and the mth nonzero gene cannot be the same since any AIR should not visit the same base placement more than once.
4. If the deployed AIRs are identical, then it should not be possible to have $g_{ik} = g_{jm}$, i.e. it should not be allowed to have the kth nonzero gene in the ith part of a chromosome be the same as the mth nonzero gene in the jth part of a chromosome. Having more than one identical AIR visiting the same base placement will not improve the result. However, if different AIRs are deployed, then each AIR would have a different capacity, and there can be potential for better performance or a greater coverage by another AIR visiting the same base placement.

3.3.3 Mutation Operator

An example mutation operator is presented in this section, and the requirements that need to be taken into account for achieving feasible solutions are explained. The purpose of the mutation operator is to maintain diversity from one population to the next, and this is done by altering the values of a small number of genes in a chromosome [29]. For the problem under consideration, only the nonzero genes are allowed to be altered so as to avoid increasing or reducing the number of FBPs each AIR needs to visit as part of the mutation operation. The genes are altered by a small magnitude, and thus the aim is to potentially improve the performance of a chromosome by slightly changing the values (i.e. choosing neighboring FBPs) of a few nonzero genes. Fig. 10 shows an example of the mutation operation where for each part, a random number of genes of the chromosome are slightly altered by adding a small positive or negative random integer to the genes.

The number of nonzero genes to be selected and the selection of the genes can, for example, be arbitrary. Note that points 3 and 4 of the requirements mentioned for the crossover operator (i.e. in Section 3.3.2) are also applicable to the mutation operator. A small value can be added to, or subtracted from, the kth selected nonzero gene of the ith part of the chromosome. In doing so, the lower and upper bounds are not to be

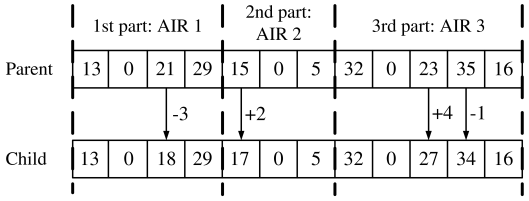


Fig. 10 An example of the mutation operation where small positive or negative random integers are added to a small number of genes

violated, i.e. the following condition is to be kept true: $1 \leq g_{ik} + \delta^s_{ik} \leq n^F_i$ where δ^s_{ik} is a small negative or positive integer added to g_{ik}, and n^F_i is the number of FBPs associated with the ith AIR. This constraint prevents the mutated gene from having a value of zero (which means one less base placement would be visited), or to have a value greater than n^F_i.

4 Experiments

This section first presents an insight into the procedure used for evaluating the objective functions, followed by an explanation of the simulated and the real-world experiments with brief results. Then, Section 5 (Results and Discussion) provides a more detailed explanation of the results, studies on repeatability and solution quality, validation of the simulation results using real AIRs, and comparisons between different solutions from the Pareto front. The simulated experiment uses real data obtained from part of a steel bridge maintenance site, and the aim is to test the OMBP approach using three AIRs. The real-world experiment is conducted using two real AIRs and a vehicle that is targeted for grit-blasting.

The function 'gamultiobj', which is based on NSGA-II, from the Matlab 2013 optimization toolbox was used for the experiments. Based on a brief investigation, it was found that the default parameters for Matlab 'gamultiobj' function perform well for the problem under consideration. Detailed investigation on the effects of different parameters, and comparing between the various optimization algorithms, is outside the scope of this paper. The computer used to run the algorithm has the following specifications: 2.8GHz Intel Xeon E5-2680 v2 and 256GB 1866MHz ECC DDR3-RAM. However, the code is single threaded and hence only uses 1 core of the CPU at any one time.

The experiments consider the grit-blasting application, which is similar to many other surface preparation manufacturing applications, such as spray-painting, surface coating, and surface polishing. The AIRs are made up of a Neobotix MP700 base, a 6 DOF Schunk industrial robot, a grit-blasting nozzle, and an RGB-D camera attached to the nozzle. The following values are used: (i) the constant end-effector speed of the AIRs is set to 0.1 meters per second for the simulated experiment, and 0.056 meters per second for the real-world experiment, (ii) the size of the targets representing the objects is set to 0.04 meters in radius, (iii) the overlap of two adjacent targets along a path is set to approx. 30% of their diameter, (iv) the threshold δ used in Eq. (9) is set to 1 meter, and (v) the set-up time, t^s_i used in Eq. (3) is set to be 10 minutes for the intended application.

To find feasible AIR poses, a lookup table the same as the one mentioned in [18] was used in the experiments to potentially reduce the computation time. The aim of finding feasible AIR poses for the targets is not to generate a trajectory for the AIR for task execution, but rather to simply examine which targets are reachable at a particular base. Thus, the use of a lookup table is effective and time efficient.

4.1 Objective Functions Evaluation

Various multi-objective optimization algorithms can be used to solve the proposed mathematical model. However, regardless of the optimization algorithm used, the objective functions are required to be evaluated iteratively within the optimization solver. Function 1 is shown and explained in this section so as to give an insight into the procedure used in the experiments for evaluating the objective functions within the employed NSGA-II optimization algorithm. The presented fitness function is specific to the application being considered; however its applicability can extend to a wide range of similar applications or scenarios.

The inputs to the fitness function (Function 1) are the design variables Z and all targets, $O = \{O_i, \ldots, O_n\}$ associated with all AIRs. The design variables are encoded as the genes of the chromosomes used in the multi-objective GA. For example, the design variables associated with the ith AIR correspond to the nonzero genes in the ith part of the chromosome. The function loops n^v times (line 3) where n^v is the number of nonzero genes in a chromosome (i.e. the total number of base placements all AIRs need to visit). At each loop, the aim is to evaluate the performance (line 13) of an AIR at one of its assigned base placements. To do so, the progress times, $t_1, t_2, \ldots, t_n$, of the n AIRs are first sorted from the lowest to the highest value (line 4) where $T^s = \{T^s_1, T^s_2, \ldots, T^s_n\}$ with corresponding indices $I^s = \{I^s_1, I^s_2, \ldots, I^s_n\}$, and $T^s_i \in \{t_1, t_2, \ldots, t_n\}$ with corresponding index I^s_i. The progress time of an AIR is the time expected to have been taken by the AIR while carrying out the task up to it current base placement (i.e. current loop). Based on the sorted progress times T^s and corresponding indices I^s, the AIR with the minimal progress time (i.e. the ith AIR in line 7) moves to its next assigned base placement. However, only if the number of base placements assigned to the AIR has not exceeded (line 9) where n^v_i is the total number of base placements the ith AIR needs to visit. If the base placement $\boldsymbol{\beta}^{AIR}_i$ that the ith AIR will move to is close to another AIR's base placement $\boldsymbol{\beta}^{AIR}_j (j \in 1, \ldots, n, j \neq i)$ (i.e. the distance checking shown in line 10 where δ is the threshold used in Eq.

Function 1 Objective Functions Evaluation

```
1: function FitFunc(Z, O)
2:     k_i ← 1, ∀i : i = 1, ..., n
3:     for Counter^base = 1 to n^v do
4:         [T^s, I^s] ← Sort{t_1, t_2, ..., t_n}
5:         í ← 1
6:         while í ≤ n do
7:             i ← (I^s)_í
8:             k ← k_i
9:             if k ≤ n_i^v then
10:                if ||β_i^AIR − β_j^AIR|| ≤ δ then
11:                    t_i = t_j
12:                end if
13:                [O_i^al, T_i^al, W_i^al, t_i] ← Perf(O, Z_ik)
14:                t_i = t_i + t_i^s
15:                k_i ← k_i + 1
16:                í ← n + 1
17:            else if k_i > n_i^v then
18:                í = í + 1
19:            end if
20:        end while
21:    end for
22:    F_1 ← Coverage(O^al, O)
23:    F_2 ← Makespan(t_1, ..., t_n)
24:    F_3 ← Manipulability(W^al)
25:    F_4 ← Torque(T^al)
26:    return [F_1, F_2, F_3, F_4]
27: end function
```

Function 2 Evaluate Performance at a Base Placement

```
1: function Perf(O, Z_ik)
2:     for Counter^rot = 1 to n^r do
3:         for j = 1 to n_i^O do
4:             if Reachable(o_ikj, Z_ik) = true then
5:                 O_i^al ← O_i^al ⌢ o_ikj
6:                 W_i^al ← W_i^al ⌢ W(q_ikj^f)
7:                 T_i^al ← T_i^al ⌢ T^Rmax(q_ikj^f)
8:                 t_i ← t_i + t(o_ikj)
9:             end if
10:        end for
11:    end for
12:    return [O_i^al, T_i^al, W_i^al, t_i]
13: end function
```

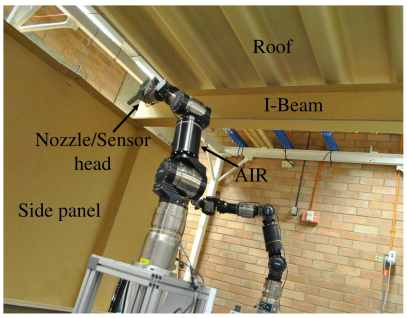
(9)), then the ith AIR is made to wait until the jth AIR has completed the work at its current base placement (line 11). The fitness function is designed based on the "first come, first served" strategy, such that when an AIR moves to its next assigned base placement, it is allocated all the targets that it can reach and that are not yet allocated to another AIR. The performance at the next assigned base placement of the ith AIR is then evaluated (line 13 - further explained below) and the progress time t_i of the AIR is updated (line 14) by adding the set-up time t_i^s of the ith AIR to the progress time. However, in the case that the limit is reached in terms of the number of base placements to be visited by the ith AIR (line 17), then the next AIR with minimal progress time, i.e. $(i+1)$th AIR, is checked to be used (lines 18).

The output data obtained from the function **Perf** (in line 13) can be used to obtain the values of objectives F_1 to F_4 (lines 22 to 25) based on Eqs. 1, 2, 5, and 8, respectively. For all the experiments, it is assumed that the AIRs need to obtain a uniform coverage of all the surfaces, thus the completion time of an AIR is based on Eq. (3).

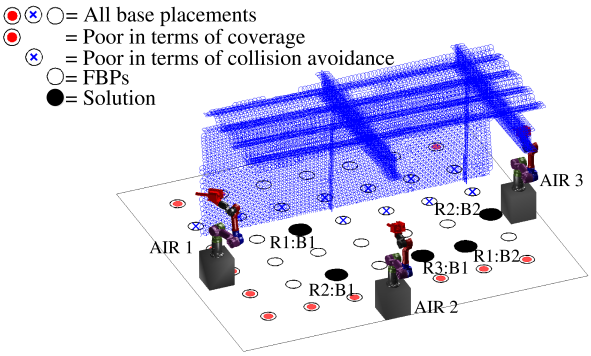
The function **Perf** (Function 2), which was used in line 13 of Function 1, is to evaluate the performance of an AIR at a particular base placement. The inputs to the function are the set of targets O, and Z_{ik} which is the design variable indicating the index of the FBP. Based on the design variable Z_{ik}, the targets O_{ik} that are inside the workspace boundary of the ith AIR at the kth base placement can be obtained. An appropriate number of discrete base rotations (i.e. n^r in line 2), which an AIR needs to perform in order to cover all targets at a particular base placement, is to be predetermined based on the structure and the kinematics of the AIR. At the kth base placement, the function loops through the base rotations (line 2) and all targets $\boldsymbol{o}_{ikj} \in O_{ik}, \forall j : j = 1 \ldots n_i^O$ (line 3). For each target, if the target is reachable (line 4), i.e. if a feasible AIR pose $\boldsymbol{q}_{ikj}^f$ can be found for the target $\boldsymbol{o}_{ikj}$, then the target $\boldsymbol{o}_{ikj}$, the manipulability measure $W(\boldsymbol{q}_{ikj}^f)$ (calculated based on Eq. (6)) due to the AIR's pose $\boldsymbol{q}_{ikj}^f$, and the maximum torque ratio $\mathcal{T}^{Rmax}(\boldsymbol{q}_{ikj}^f)$ (calculated based on Eq. (7)) are added (lines 5 to 7) to the sets O_i^{al}, $\mathcal{T}_i^{al}$ and W_i^{al}, respectively. Note that the notation $\frown$ represents concatenation. The subscript al is used to symbolize the *allocated* targets. The time it takes to cover the target $\boldsymbol{o}_{ikj}$ (i.e. $t(\boldsymbol{o}_{ikj})$) is also added to the progress time t_i of the ith AIR (line 8).

4.2 Three AIRs Applied in a Steel Bridge Maintenance Environment

A mock environment as shown in Fig. 11a is created using real data obtained from a part of a steel bridge maintenance site. Simulations were then performed on the data obtained from the mock environment (Fig. 11b). Three identical simulated AIRs modeled upon real AIRs are used to perform the task of grit-blasting. The simulated scenario consists of 7518 targets (shown as small blue disks in Fig. 11b) to represent all the surfaces to be cleaned in the environment. The circles on the ground are all the discrete base placements, from which the 18 empty circles are the FBPs. When determining the FBPs, in order to discard the base placements that have low coverage of the targets representing the objects, an estimation for coverage can quickly be made by calculating the number of targets that fall inside the workspace boundary of an AIR at each of the discrete base placements. This estimation can significantly reduce the computation time since feasible AIR poses are not generated for assessing the reachability of



(a) The environment in which scanning was performed to obtain the data



(b) The result of the simulation

Fig. 11 AIRs applied for steel bridge maintenance

the targets. However, after determining the FBPs and during the optimization process, accurate measures of coverage need to be made based on the aforementioned lookup table. Note that although only the internal surfaces of the structure need to be covered, there are discrete base placements generated at the rear of the structure, since some of the targets (such as the back-end of the roof) can only be reached from the rear.

Based on the designed selection strategy (explained in the next section), a solution from the Pareto front is selected which is shown in Fig. 11b where the annotated and filled black circles are the selected base placements for the AIRs. The notation Ri:Bj represents the ith AIR at its jth base placement to visit. Based on the chosen solution, 96.5% of the reachable targets can be covered. Solutions with over 99% coverage can be obtained from the Pareto front; however at the cost of a significant increase to the makespan, as will be explained in the next section.

4.3 Real-World Experiment Based on Two AIRs Deployed to Perform Grit-blasting on a Vehicle

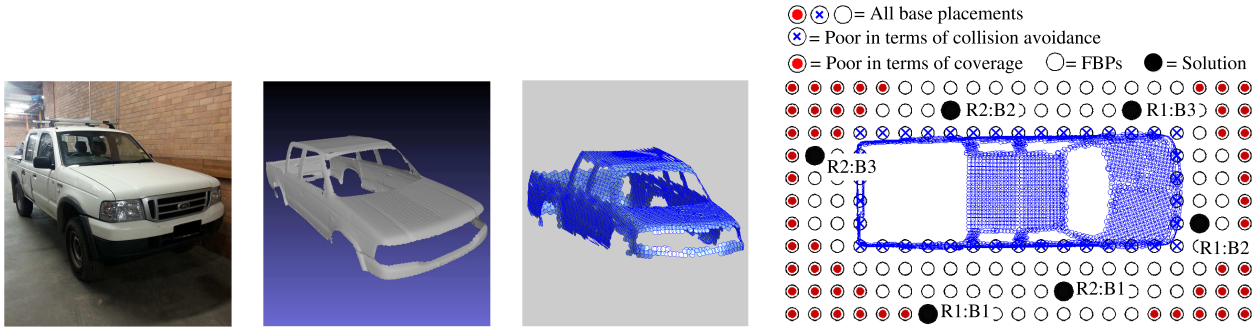
As shown in Fig. 12, a real-world experiment using two AIRs and a vehicle (utility truck) is carried out. The experiment resembles the one-off task of grit-blasting of vehicles for removing of old paint from metallic surfaces as a preparation for new paint. After scanning and processing the scan data, the point cloud that is shown in Fig. 12b can be obtained, which represents the metallic surfaces of the vehicle. The point cloud can then be used for target representation of the vehicle as shown in Fig. 12c where 3270 targets (shown as the blue disks) are used to represent the surfaces. In Fig. 12d, the circles on the ground are all the discrete base placements, from which the 87 empty circles (including the black filled circles) are the FBPs. The annotated and filled black circles are the chosen base placements for the AIRs.

Solutions with over 99% coverage can be obtained from the Pareto front. However, based on the designed selection strategy, a solution from the Pareto front is selected with 94% coverage of the reachable targets and a makespan of 3126 seconds. The difference between the completion times of the AIRs is only 46 seconds (less than 1.5%).

5 Results and Discussion

In this section, a discussion is made on the selection of a solution from the Pareto front for the grit-blasting application, followed by detailed comparisons and evaluations of the results. Discussions on the parameters effecting the computation time and the calculation of feasible AIR poses are also presented.

The selection strategy to be designed for selection of a solution from the Pareto front is highly dependent on the application being considered. For the autonomous and one-off grit-blasting scenarios considered in the experiments, achieving an above threshold coverage is vital. Hence, a small subset of solutions are selected from the Pareto front such that an acceptable coverage percentage of the surfaces is obtained, i.e. narrowing down the solutions based on objective 1. Note that for some applications 100% coverage is necessary. Hence, the selection strategy in these applications must only select the solutions with 100% coverage. From this subset of solutions, a further subset is chosen based on the makespan (objective 2). Finally, the weighted average of the manipulability measure (objective 3) and the torque (objective 4) can be the basis of choosing the final solution from the reduced subset of solutions. Alternatively, the final solution can be selected based on improving the manipulability of the AIRs (i.e. objective 3) or minimizing the torque experienced by the AIRs (i.e. objective 4) if the joints condition of the AIRs is more critical.



(a) A photo of the vehicle **(b)** Point cloud and triangle mesh **(c)** Target representation of the vehicle **(d)** A solution chosen from the Pareto front

Fig. 12 The vehicle, the point cloud representation and the target representation

5.1 Evaluation of the Simulated Experiment

In Table 2, three solutions from the Pareto front are presented. For clarity, solutions related to Objective 1 are presented as the coverage percentage of the *reachable* targets rather than missed-coverage. As shown in the table, 99% coverage of the reachable targets is possible. However, assuming a coverage threshold of 96% for this scenario, and based on the selection strategy describe previously, then solution 3 would be the final solution (Fig. 11b is based on this solution). Solution 3 has a significantly better makespan than the other two solutions. Of course, the selection strategy can be changed to suit the desired expectations for the application. As explained in Section 3.1, finer base placement discretization will be needed if the desired threshold is not met. It needs to be noted that a maximum of 93.5% of the targets that represent the object can actually be covered based on a brute force search used to obtain the ground truth. The rest of the targets, representing areas such as the top and the bottom of the I-beam flange and the back end of the roof, cannot be covered regardless of the base placement of the AIRs (unless the AIRs are equipped with a base that can move vertically). Hence, 96.5% of *reachable* targets correspond to 90.2% coverage of the entire object. Note that the optimizer minimizes objectives functions; however, objective 3 (F_3), which is related to manipulability measure, has to be maximized. Thus, the optimizer minimizes $-F_3$.

Table 2: Three solutions from the Pareto front for the simulated experiment

Soln. #	Obj. 1 (% coverage)	Obj. 2 (makespan in sec.)	Obj. 3 (manipulability)	Obj. 4 (Torque in N.m)
1	99	3328	-459	1991
2	98.4	3261	-435	2565
3	96.5	2505	-374	1724

5.2 Checking Solution Quality and Consistency

Tuning of optimization parameters have been briefly investigated and the default parameters for Matlab's 'gamultiobj' function were found to perform well for the problem under consideration. However, comparing between different optimization algorithms that can solve the mathematical model and a detailed investigation in further tuning the optimization algorithm's parameters are left for future work. Henceforth is a demonstration that NSGA-II and the developed chromosome representation are suitable for solving the proposed mathematical model. Solution quality and consistency in obtaining acceptable solutions are checked by repeating the optimization process for the simulated experiment (10 times) and then evaluating the results.

Fig. 13 shows the results for objective 1 (percentage of missed-coverage) and objective 2 (makespan in

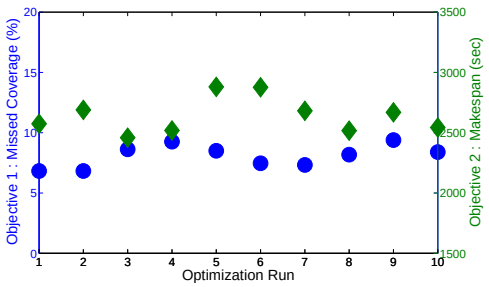
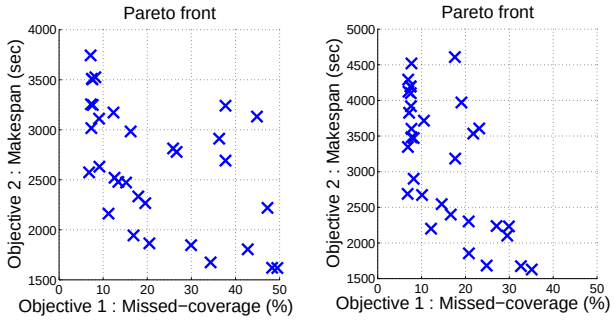


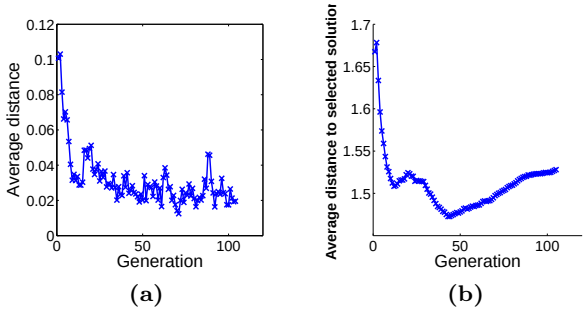
Fig. 13 Results for objective 1 (percentage of missed-coverage) and objective 2 (makespan in seconds) of the 10 optimization runs



(a) Pareto front for the first optimization run

(b) Pareto front for the second optimization run

Fig. 14 Pareto fronts for the first two optimization runs with respect to objectives 1 and 2 only



(a) **(b)**

Fig. 15 (a) Average of distances of individuals at each generation; and (b) average of distances of all individuals in each generation to the selected solution

seconds) of the 10 optimization runs. The solutions are selected based on the same selection strategy used before and all solutions satisfy the 96% coverage threshold of the reachable targets. Fig. 13 shows that a near-optimal solution is found by each optimization run, hence demonstrating the robustness of NSGA-II and the developed chromosome representation for the problem under consideration. The average of the 10 solutions for objectives 1 to 4 is 8.1% missed-coverage (corresponds to 98% coverage of the reachable targets), 2641 sec, -417 and 1605 N.m, respectively.

Figs 14a and 14b show the Pareto front for the first 2 optimization runs. It can be seen that there are a number of solutions that meet the 96% coverage threshold (less than or equal to 10% missed-coverage). Note that although the Pareto front is shown with respect to objectives 1 and 2, the Pareto front actually considers all four objectives, hence some of the solutions are better in terms of objectives 3 and/or 4 rather than objectives 1 and 2. It was found that in all 10 optimization runs, the optimization terminates due to the average change in the spread of the Pareto front being less than the default tolerance set in 'gamultiobj' function of the Matlab optimization toolbox. The default maximum number of generations calculated by the optimization solver based on the number of design variables is 1200; however, on average optimization terminates at 101 iterations.

Figures 15a and 15b are created from an optimization run. Figure 15a shows the average of distances of individuals at each generation using the field 'AverageDistance' of Matlab 'gamultiobj', which tends to favorably decrease as the generation number increases. At each generation, the average of distances of each member of the population to the nearest neighboring member is calculated and saved to the field 'AverageDistance'. Figure 15b shows the average of distances of all individuals in each generation to the selected solution from the Pareto front which was presented in Section 5.1. The individuals in each generation were normalized with respect to the selected solution. It can be seen in Fig. 15b that for the first 45 generations, the average of distances decreases with respect to the selected solution. However, the average value then starts to slowly increase which could be due to the optimizer attempting to further diversify the population in the hope of arriving at better solutions. These solutions may also be better with respect to other solutions in the Pareto front.

The aim of the above investigations is to ensure consistency and to test that the solutions are appropriate for the problem under consideration. However, this information can also be used for future studies as a means to investigate faster convergence (e.g. using a hybrid optimization) and to improve computation efficiency.

5.3 Evaluation and Comparative Study of the Real-world Experiment

In Section 4.3, an experiment was presented where two mobile AIRs were deployed to carry out the task of grit-blasting on a vehicle (utility truck). Based on the designed selection strategy presented previously and assuming a coverage threshold of 90% for this scenario (one-off grit-blasting of a vehicle), a solution from the Pareto front is selected (Fig. 12d). The solution provided values of 20%, 3126 seconds, -186, and 507 N.m for objectives 1 to 4, respectively. Note that 15% of the targets, which mostly represent the roof of the vehicle, can't be covered by any AIR regardless of the chosen base placement since these targets are too high relative to the base of the AIRs. Thus, although the value of objective 1 for the selected solution is 20% (meaning that

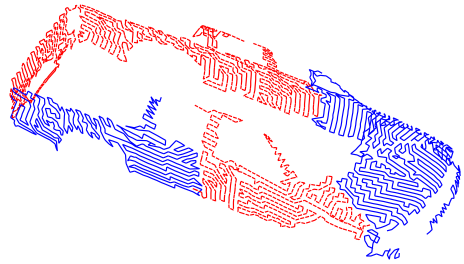
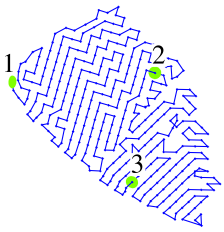


Fig. 16 Paths of AIRs 1 (blue lines) and 2 (red lines)

20% of the total target were not covered), in reality only 6% ($1-0.8/0.85$) of the *reachable* targets were not covered. Based on the selected solution, the completion time of AIRs 1 and 2 is expected to be 3088 and 3126 seconds, respectively. A solution with over 99% coverage can be obtained; however at a significant cost to makespan. This is because the set-up time, t_i^s is set to a conservative value (10 minutes), and to cover a slightly larger percentage of the reachable targets, the number of base placements to be visited by at least one of the AIRs needs to be increased. In some applications (e.g. autonomous and one-off grit-blasting of objects), it can be more convenient and time efficient for the missed out sections to be covered manually by a human operator; otherwise, the selection strategy needs to consider a 100% coverage threshold.

Fig. 16 is created to illustrate the areas of the vehicle that are reachable based on the selected solution by showing the paths generated for both AIRs where the paths shown as blue lines and red lines are associated with AIRs 1 and 2, respectively. The feasibility of the obtained solution and the paths generated for both AIRs are checked using the AIRs. As an example, the path that is generated on the bonnet of the vehicle, which is associated with the first AIR at its second base placement, is shown in Fig. 17a. The experiment set-up at this base placement is shown in Fig. 17b. To check for the correct coverage of the path by the AIR, a laser is installed at the end-effector of the AIR and the laser point is tracked and compared to the simulated path. The highlighted targets in the path shown in Fig. 17a correspond to the laser point locations shown in Fig. 17c to 17e. The actual completion time of AIRs 1 and 2 is found to be 3080 and 3111 seconds, respectively. It is sometimes not possible to generate paths that will appropriately and completely cover all reachable targets, and as a result, a small difference in completion times between the simulation result and the experiment can be present. However, the difference is insignificant (less than 15 seconds or 0.5%) for this experiment.



(a) Three highlighted targets on a path

(b) Set-up of the experiment

(c) Laser point at the 1st target

(d) Laser point at the 2nd target

(e) Laser point at the 3rd target

Fig. 17 The coverage of the path associated with the first AIR at its second base placement is checked using a laser that is installed at the end-effector of the AIR

Similar to the simulated experiment, to check for consistency and to test that the employed optimization algorithm can produce near-optimal results every time it is run, the optimization process is repeated 10 times for this experiment using the same procedure outlined in Section 5.2. A near-optimal solution is found each time and the average of the 10 solutions for objectives 1 to 4 is 21.6% (corresponds to 92.2% coverage of reachable targets), 3170 sec, -181 and 498 N.m, respectively.

5.4 Summary of Main Results

For the steel bridge maintenance environment where three AIRs were deployed, on average, 98% coverage of the reachable targets was achieved. Similarly, for the grit-blasting experiment carried out on a vehicle using two real AIRs, on average, 92.2% coverage of the reachable targets was achieved with near-optimal makespan (based on the selection strategy designed and the coverage threshold considered for each scenario). For both experiments, a coverage percentage greater than 99% could be achieved; however at the cost of a substantial increase in the makespan. Note that space discretization does effect the performance, hence if the desired threshold is not achieved, then finer discretization of the search space is needed as per the explanation in Section 3.1. The optimization process was repeated 10 times for both experiments to ensure consistency in achieving acceptable results, and to check the suitability of using NSGA-II and the developed chromosome representation for the problem. Hence, the solutions are selected from the Pareto front such that the coverage percentage

satisfies the threshold considered for the scenario while providing a near-optimal makespan.

5.5 Computation Time

Table 3 provides the off-line computation time of the experiments for running the optimization (average of 10 runs). For comparisons, the number of targets representing the objects, and the number of FBPs evaluated are also provided. Although the code is single threaded, it is recognized that a multi-threaded implementation could potentially lead to a significant reduction in computation time.

There is room for improving the computation efficiency, and further research that focuses on the efficiency of the approach can be carried out. For example, a benchmark for assessing the discretization of the search space can be developed with the aim of reducing the computation time. Detailed studies on the tuning of parameters that effect the multi-objective GA can also be made to compare time efficiency against solution quality for the target application. Moreover, the objects can be represented with fewer targets, meaning that fewer AIR poses need to be found, which is predicted to reduce the computation time.

6 Conclusions

An approach was presented to address the problem of effective collaboration of multiple autonomous industrial robots to perform manufacturing tasks such as grit-blasting, spray-painting and other surface preparation tasks. More specifically, the problem of collaboration through optimal base placements of the autonomous robots that are equipped with manipulators was investigated. The presented approach enabled each robot to simultaneously determine: (i) an appropriate number of base placements, (ii) the location of the base placements, and (iii) the visiting sequence of the chosen base placements, such that complete coverage of the surfaces is obtained. The approach included search space discretization and candidate base placements selection, followed by a mathematical model that takes into account multiple objectives (i.e. maximal coverage, minimal makespan, maximal manipulability measure and minimal AIRs' joints torque). A minimum distance constraint between robots is enforced inherently by the mathematical model. Simulated and real-world experiments were conducted to compare the objectives and to validate the approach.

Future work includes finding optimal base placements by accounting for the various uncertainties inherent to the environment. Methods to improve the computation time can also be studied. Developing criteria for measuring the performance of the space discretization would also be of interest. Variations of the approach can also be studied, e.g. by accounting for the presence of immobile AIRs when optimizing coverage for the mobile AIRs; and incorporating AIR path planning parameters, such as the actual path traversal time of the base, while utilizing Spatio-temporal optimization.

Table 3: Computation time for the experiments

	Simulated Experiment	Real-world Experiment
Computation time	15.3 mins	5.9 mins
# of targets	7518	3270
# of base placements	18	87

Acknowledgements This research is supported by SABRE Autonomous Solutions Pty Ltd and the Centre for Autonomous Systems (CAS) at the University of Technology Sydney, Australia. A special thank to Dr. Andrew Wing Keung To for his feedback and assisting with various aspects of the experiments. Authors also thank Prof. Gamini Dissanayake, Assoc. Prof. Shoudong Huang, Mr. Teng Zhang and Mr. Raphael Falque for their valuable recommendations and discussions.

References

1. Hvilshoj, M., Bogh, S., Skov Nielsen, O., Madsen, O.: Autonomous industrial mobile manipulation (AIMM): past, present and future. Industrial Robot: An International Journal, 39(2), 120–135 (2012)
2. Paul, G., Webb, S., Liu, D., Dissanayake, G.: Autonomous robot manipulator-based exploration and mapping system for bridge maintenance. Robotics and Autonomous Systems, 59(78), 543–554 (2011)
3. Paul, G., Liu, D., Kirchner, N., Dissanayake, G.: An effective exploration approach to simultaneous mapping and surface materialtype identification of complex three-dimensional environments. Journal of Field Robotics, 26(11/12), 915–933 (2009)
4. Hassan, M., Liu, D., Huang, S., Dissanayake, G.: Task oriented area partitioning and allocation for optimal operation of multiple industrial robots in unstructured environments. In: 13th International Conference on Control Automation Robotics Vision (ICARCV), pp. 1184–1189 (2014)
5. Clifton, M., Paul, G., Kwok, N., Liu, D., Wang, D.-L.: Evaluating performance of multiple RRTs. In: International Conference on Mechtronic and

Embedded Systems and Applications, pp. 564–569 (2008)
6. Montiel, O., Sepúlveda, R., Orozco-Rosas, U.: Optimal path planning generation for mobile robots using parallel evolutionary artificial potential field. Journal of Intelligent & Robotic Systems, 79(2), 237–257 (2015)
7. Galceran, E., Carreras, M.: A survey on coverage path planning for robotics. Robotics and Autonomous Systems, 61(12), 1258 – 1276 (2013)
8. Gautam, A., Jha, B., Kumar, G., Murthy, J. K., Ram, S. A., Mohan, S.: Fast. Journal of Intelligent & Robotic Systems, pp. 1–20 (2016)
9. Hameed, I. A.: Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain. Journal of Intelligent & Robotic Systems, 74(3), 965–983 (2014)
10. Hsu, P.-M., Lin, C.-L., Yang, M.-Y.: On the complete coverage path planning for mobile robots. Journal of Intelligent & Robotic Systems, 74(3), 945–963 (2014)
11. Aly, M. F., Abbas, A. T., Megahed, S. M.: Robot workspace estimation and base placement optimisation techniques for the conversion of conventional work cells into autonomous flexible manufacturing systems. International Journal of Computer Integrated Manufacturing, 23(12), 1133 – 1148 (2010)
12. Yang, J. J., Yu, W., Kim, J., Abdel-Malek, K.: On the placement of open-loop robotic manipulators for reachability. Mechanism and Machine Theory, 44(4), 671 – 684 (2009)
13. Sotiropoulos, P., Aspragathos, N., Andritsos, F.: Optimum docking of an unmanned underwater vehicle for high dexterity manipulation. IAENG International Journal of Computer Science, 38(1), 48 – 56 (2011)
14. Sotiropoulos, P., Tosi, N., Andritsos, F., Geffard, F.: Optimal docking pose and tactile hook-localisation strategy for AUV intervention: The DIFIS deployment case. Ocean Engineering, 46(0), 33 – 45 (2012)
15. Mitsi, S., Bouzakis, K. D., Sagris, D., Mansour, G.: Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm. Robotics and Computer-Integrated Manufacturing, 24(1), 50–59 (2008)
16. Vosniakos, G.-C., Matsas, E.: Improving feasibility of robotic milling through robot placement optimisation. Robotics and Computer-Integrated Manufacturing, 26(5), 517–525 (2010)
17. Boschetti, G., Rosa, R., Trevisani, A.: Optimal robot positioning using task-dependent and direction-selective performance indexes: General definitions and application to a parallel robot. Robotics and Computer-Integrated Manufacturing, 29(2), 431–443 (2013)
18. Hassan, M., Liu, D., Paul, G., Huang, S.: An approach to base placement for effective collaboration of multiple autonomous industrial robots. In: International Conference on Robotics and Automation (ICRA), pp. 3286–3291 (2015)
19. Carlone, L., Kaouk Ng, M., Du, J., Bona, B., Indri, M.: Simultaneous localization and mapping using rao-blackwellized particle filters in multi robot systems. Journal of Intelligent & Robotic Systems, 63(2), 283–307 (2011)
20. Parker, L. E.: Multiple mobile robot systems. In: Siciliano, B., Khatib, O. (eds.), Springer Handbook of Robotics, pp. 921–941, Springer Berlin Heidelberg (2008)
21. Patel, S., Sobh, T.: Manipulator performance measures - a comprehensive literature survey. Journal of Intelligent & Robotic Systems, pp. 1–24 (2014)
22. Yoshikawa, T.: Manipulability of robotic mechanisms. The International Journal of Robotics Research, 4(2), 3–9 (1985)
23. Niku, S.: Introduction to robotics: Analysis, Control, Applications. John Wiley & Sons (2010)
24. Marler, R., Arora, J.: Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization, 26(6), 369–395 (2004)
25. Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., Zhang, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation, 1(1), 32 – 49 (2011)
26. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J., Schwefel, H.-P. (eds.), Parallel Problem Solving from Nature PPSN VI, vol. 1917 of Lecture Notes in Computer Science, pp. 849–858, Springer Berlin Heidelberg (2000)
27. Carter, A. E., Ragsdale, C. T.: A new approach to solving the multiple traveling salesperson problem using genetic algorithms. European Journal of Operational Research, 175(1), 246 – 257 (2006)
28. Ghosh, S. K.: Approximation algorithms for art gallery problems in polygons. Discrete Applied Mathematics, 158(6), 718 – 722 (2010)
29. Srinivas, M., Patnaik, L.: Genetic algorithms: a survey. Computer, 27(6), 17–26 (1994)
30. Whitley, D.: A genetic algorithm tutorial. Statistics and Computing, 4(2), 65–85 (1994)