Hybrid type multi-robot path planning of a serial manipulator and SwarmItFIX robots in sheet metal milling process

Satheeshkumar Veeramani¹ · Sreekumar Muthuswamy¹

Received: 19 April 2021 / Accepted: 7 August 2021 / Published online: 28 August 2021
 © The Author(s) 2021

Abstract

This work investigates on the coordinated locomotion between a ceiling-mounted serial manipulator and two SwarmItFIX robots. The former holds the machining tool as an end effector, and the other two robots act as swarm robotic fixtures in a sheet metal milling process. A novel offline coordination planner which follows the hierarchical based hybrid type decentralized planning strategy has been proposed. Motion of the serial manipulator and SwarmItFIX robots' coordinated locomotion are divided into three sub-problems, viz, trajectory planning of serial manipulator, task planning of SwarmItFIX robots, and homogenous prioritized multi-robot path planning of SwarmItFIX robots. Mathematical formulation of all the three sub-problems is developed and presented in this paper. A hexagonal segment that fits inside the boundaries of the workspace is considered as the machining trajectory. The tool velocity is assumed to be constant as it improves the quality of machining. The results obtained from the proposed planner is found to be efficient as the task planning module has computed the precise support locations and support duration for the SwarmItFIX robots. The multi-robot path planning module of the planner computes the optimal collision-free paths of SwarmItFIX robots for all goal positions. Finally, trajectories of SwarmItFIX robots are found to be completely in-line with the trajectory of tool center point (TCP) of the serial manipulator.

Keywords Robot-assisted sheet milling · Prioritized reinforcement learning · Multi-robot coordination · Multi-robot path planning · SwarmItFIX robots

Abbreviations

DH	Denavit Hartenberg
MDP	Markov decision process
MRROC++	Multi robot research oriented controller of SwarmItFIX robots
PDL	Perl data language
PKM	Parallel kinematic machine
RFA	Robot fixtureless assembly
RL	Reinforcement learning
SaD	Swing and Dock
SARSA TD	State action reward state action temporal differencing
SwarmItFIX	Swarm intelligent fixtures
TCP	Tool center point

List of symbols

${}^{pkm}\theta_{j,m}$	Angle of the j th joint of pkm agent for the m th support location
${}^{pkm}\theta_{j,park}$	Angle of the j th joint of pkm agent for the parking location
$\left. \begin{matrix} {}^{sm}\theta_{k,w-1} \\ {}^{sm}\theta_{k,w} \end{matrix} \right\}$	Angle of the k th joint of serial manipulator for the waypoints $w-1$ and w
${}^{sm}\theta_{k,home}$	Angle of the k th joint of serial manipulator for the home position
$\left. \begin{matrix} {}^{sm}\theta_k^+ \\ {}^{sm}\theta_k^- \end{matrix} \right\}$	Upper and lower angular limits of k th joint of serial manipulator
$\left. \begin{matrix} {}^{sm}\theta_k, {}^{sm}d_k \\ {}^{sm}a_k, {}^{sm}\alpha_k \end{matrix} \right\}$	DH parameters of the serial manipulators ($k=0-6$)
η	Difference in angular configuration of serial manipulator between two consecutive waypoints
${}^{tool}v$	Tool feed velocity

✉ Sreekumar Muthuswamy
 msk@iiitdm.ac.in

¹ Centre for AI, IoT, and Robotics, Department of Mechanical Engineering, Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram, Chennai 600127, India

$\left. \begin{matrix} \text{vert } x_n \\ \text{vert } y_n \\ \text{vert } z_n \end{matrix} \right\}$	Coordinates of the n th vertex of the trajectory	$R(s,a)$	Reward obtained by the computational agent by performing action a at state s
$\text{head } \theta_{m,n}$	Orientation of head agent for the m th support location belongs to n th segment.	$Q(s,a)$	Q Value of the current state-action pair
x_0, y_0, z_0	Coordinates of the point of origin	$Q(s',a')$	Q Value of the target state-action pair
$\left. \begin{matrix} \text{sm } x_b \\ \text{sm } y_b \\ \text{sm } z_b \end{matrix} \right\}$	Coordinates of the base frame (b) of the serial manipulator	α	Step size parameter
$\left. \begin{matrix} \text{sm } x_w \\ \text{sm } y_w \\ \text{sm } z_w \end{matrix} \right\}$	Coordinates of the w th waypoint of the serial manipulator trajectory	ϵ	Parameter to balance exploration and exploitation of the computational agent
$\left. \begin{matrix} \text{head } x_{m,n} \\ \text{head } y_{m,n} \\ \text{head } z_{m,n} \end{matrix} \right\}$	Coordinates of the m th support location of n th segment of the head agent	γ	Discount factor
$\left. \begin{matrix} \text{head } x_{m-1,n} \\ \text{head } y_{m-1,n} \\ \text{head } z_{m-1,n} \end{matrix} \right\}$	Coordinates of the head agent previous to m th support location of n th segment ($\text{head } x_{m,n}, \text{head } y_{m,n}, \text{head } z_{m,n}$)	π	Non-optimal policy obtained from planner
ϕ_n	Angle of the n th segment of the trajectory	π^*	Sub-optimal policy obtained from planner
${}^{\text{hh}}D_n$	Distance between two adjacent head positions of the n th segment	$Rq\pi_m^*$	Computed sub-optimal policy of the q th robot for the m th task
${}^{\text{ch}}D_m$	Distance between contour and head at m th support location	${}^{\text{sm}}\text{Traj}$	Complete trajectory plan of the serial manipulator robot
R_q	q Th SWarmItFIX robot ($q = 1, 2, 3, 4$)	$Rq\text{Seq}$	Locomotion sequence of the q th robot
${}^{\text{base}}\text{Gpos}_m$	m Th Goal position of the traversing base agent	N_{pins}	Number of docking pins in the workbench ($P1, P2, \dots, P52$) in the present case
${}^{\text{base}}\text{Gpos}_m - 1$	Previous goal position to ${}^{\text{base}}\text{Gpos}_m$	N_{edge}	Possible number of edges which connects docking pins
${}^{\text{base}}\text{Cpos}_q$	Current position of the traversing base agent of q th robot	N_{state}	Possible number of docking positions in the workbench ($C1, C2, \dots, C78$). Each docking position corresponds to a set of three docking pins
${}^{\text{base}}\text{Hpos}_q$	Home position of the traversing base agent of q th robot	$\left. \begin{matrix} \vec{n}, \vec{o} \\ \vec{a}, \vec{p} \end{matrix} \right\}$	Normal, orthogonal, approach, position vectors of TCP of the serial manipulator
$RqBseq_m$	Action sequence of the base agent of q th robot for m th task		
Task_m	m Th Task for the SWarmItFIX robot		
$t_{m,n}$	Time stamp for the SWarmItFIX robot for the m th task belongs to n th segment		
$\left. \begin{matrix} T_{\text{odd}} \\ T_{\text{even}} \end{matrix} \right\}$	Odd and even tasks of Task_m		
${}^{\text{obs}}\text{Task}_m$	Obstacles for the m th task		
${}^{\text{rob}}\text{Task}_m$	Allotted robot for the m th task		
S	State space of the MDP, consist of (s_1-s_{78})		
A	Action space of the MDP, consist of (a_0-a_8)		

Introduction

SwarmItFIX [1] (Fig. 1a) is a novel robotic setup that has been developed as a robot fixtureless assembly (RFA) platform under the European Union project mainly for sheet metal manufacturing applications. However, with certain modifications, this RFA platform can contribute to other applications, including material handling and material transfer. SwarmItFIX is a multiagent platform consisting of three different agents: head, parallel kinematic machine (PKM), and Base (Fig. 2). For a single SwarmItFIX robot, all the three agents have to work in a coordinated manner to accomplish any particular task. These robots perform a novel Swing and Dock (SaD) locomotion on a fixed modular platform, called as workbench. The workspace of these SwarmItFIX platform can be modified by varying the size of the workbenches or by increasing the number of docking pins. The details of the SwarmItFIX setup and the agent coordination have been described in detail in the previous works of the authors [2–4]. A software framework called MRROC++ has been utilized for the path planning of robot systems including ASEA IRB 6 and SwarmItFIX [5].

Serial manipulators (Fig. 1b) assist CNC machines during machining which would enhance throughput. In

Fig. 1 Robot setup

Fig. 2 CAD model of the SwarmItFIX setup integrated with the serial manipulator

robot-assisted machining, for maintaining the proper balance between machining efficiency and quality, constant feed rate is generally preferred. Different feed rate segments in a single machining trajectory increase the joints' jerk, which ultimately leads to deviation in the machining trajectory [6]. The repeatability error present in the trajectory of a serial manipulator is directly proportional to its reach. This error is in the inner areas of the work and high in the areas when the robot attains its maximum reach [7, 8]. The stiffness of joints will be heterogeneous throughout the work envelope of the serial manipulator hence, the machining has to be performed within its possible range of best stiffness [9–11]. In this work, a serial robot possessing higher work volume is deployed in coordination with SwarmItFIX robots.

Given the coordinates of the target position and direction of tool, the end-effector target pose of the serial manipulator can be determined. The inverse kinematics model computes the angles of all joints for a corresponding end-effector target pose (Fig. 3). As multiple solutions are possible, optimization-based inverse kinematics model is generally preferred for the computation of joint angles. In addition, several constraints can be imposed for achieving joint angles within the required range, so that an abrupt change in joint angles between two adjacent target points could be avoided.

The Markov decision process (MDP) addresses the sequential decision problems in fully observable stochastic environments with transition and reward functions. Earlier researchers have also made the MDP to address the deterministic environments [12]. The problems modeled as MDPs are mostly solved using model-based or model-free reinforcement learning (RL) algorithms. In general, both the techniques are offline or static in their working nature, wherein an optimal or sub-optimal policy (π^*) can be obtained from the computational environment which will be utilized for the real-time problems. In both the RL algorithms, the computational agents take actions in their computational environment and collect the reward, where the objective is to maximize the cumulative reward. The major difference between the model-based and model-free RL algorithms is that, the former requires a transition model; whereas, it is not required for the other. As the SwarmItFIX robots' workspace is modular, the model-free RL techniques are preferred for the path planning of SwarmItFIX robots. In this method, a random policy (π) is fed instead of calculating a transition model. The computational agent performs action

(a) at any particular state (s) suggested by π , and attains a new state (s') and receives the reward $R(s,a)$. According to the utility matrix $Q(s,a)$ will be updated automatically. Based on this updated $Q(s,a)$, the existing policy π will be refined further. The process will be repeated for a predefined number of iterations or until the optimal or sub-optimal policy is obtained [13].

The multi-robot systems can either be homogenous (robots of the same kind) or heterogeneous (robots of a different kind) depends on their capabilities [14]. The current work deals with a novel hybrid multi-robot system, consisting of one serial manipulator robot and two SwarmItFIX robots. The serial manipulator's role is to perform the machining operation with end effector as the tool, whereas, the SwarmItFIX robots provide support from the bottom of the sheet metal. The geometrical and FEM constraints (unary and binary) help to obtain a safe distance to be maintained between two consecutive heads ($^{hh}D_n$) and between head and contour ($^{ch}D_m$) [15, 16]. In this work, for the multi-robot path planning and decision-making problems of SwarmItFIX robots, a hierarchical decentralized static or offline strategy has been proposed. The computational agents communicate with each other without a centralized agent, traverse in the computational environment, and identify the collision-free optimal path. When the number of agents increases, the complexity of the coordination problem will also increase. In those situations, this decentralized approach provides more flexibility and adaptability for the system than that of the centralized approach. Also, the offline strategy become simpler in those situations compared with the online strategy. Moreover, it is cumbersome to use the dynamic or online method in the presence of a heterogeneous multi-robot system [14].

Generally, planning in multi-robot systems will be classified into task planning and motion or path planning. Task planning decomposes the complete problem into multiple sub-tasks and designates them to various robots deployed. Simultaneously, the motion planning generates a trajectory or path for each robot to accomplish the designated task. In this work, initially, the task planning for SwarmItFIX robots and the serial manipulator robot has been done individually. Finally, the motion planning of serial manipulator robot and SwarmItFIX robots are performed using inverse kinematics and prioritized reinforcement learning, respectively.

The contribution of this paper includes the following:

- Task planning of SwarmItFIX robots.
- Modeling the multi-robot coordination problem of SwarmItFIX as MDP.
- Development and implementation of a novel prioritized SARSA algorithm to solve the proposed MDP.
- Development of optimization-based inverse kinematics model of the ceiling-mounted serial manipulator robot.

Fig. 3 Forward vs inverse kinematics

- Development of multi-robot planner for the coordinated locomotion of serial manipulator robot with two SwarmItFIX robots for sheet metal milling operation.

Related works

The work [17] addresses the position, velocity, and acceleration analysis of a ceiling-mounted 6-axis serial manipulators. The work [18] shows, by integrating deep RL and inverse kinematics models, an efficient path planning model can be obtained for the serial manipulators. However, the method is under implementation. The optimization-based inverse kinematics model successfully provides joint angles while avoiding singularities. The same optimization-based model has also been effectively implemented for the robotic milling tasks [9–11]. The literature also identifies that, more than the analytical methods like IKFast [19], optimization based methods are more suitable for solving inverse kinematics problem of serial manipulators that performs milling operations [9–11]. The gradient-based solvers are more efficient in solving non-linear optimization-based inverse kinematics problems as they allow additional constraints. This restrict sudden changes in joint variables between two consecutive way points unlike the evolutionary algorithm-based solvers such as Bio-IK [19]. Hence, in this work the objective function and the constraints are newly framed, however, the existing BFGS Gradient Projection solver has been utilized for calculating the joint angles of the ceiling mounted serial manipulator.

For multi-agent systems, it would be more appropriate that the agents compute the plan/policies individually (distributed approach) rather than relying on the centralized units. The distributed approach consists of two phases; (i) deciding the priority of the agents (task planning), and (ii) computation of plans of individual agents by considering the information of other agents (motion planning). Moreover, the agents are fed with limited information on other agents which ultimately reduces the computational complexity. Based on these aspects, the distributed approach is most preferred for the multi-robot path-planning problems [20–22]. In Ref. [23], the authors investigated two distributed algorithms, for solving three different cases of multi-agent path planning problems. The results showed that the priority-based algorithm outperforms the other distributed algorithm in all three cases regarding success rate. This priority-based algorithm also outperforms many existing centralized algorithms includes MA-A*, EPEA*, and CBS, in terms of runtime. Authors in [24] have modeled the automated valet parking problem as a multi-agent path planning problem, and solved it using two prioritized methods, constraint-based method (CBS-Pri) and prioritized heuristic method (CA*-Pri). Based on the results, it was found that

the CBS-Pri gives the optimal solution by taking more time comparatively. Whereas, CA*-Pri takes lesser time and does not guarantees the optimal solution. The authors [25, 26] have proved that the prioritized approach for the multi-agent path planning problem yields optimal results with reduced makespan with less computation time and does not impose any constraints related to communication, external coordination, and synchronization among robots. Unlike coordinated approaches, at the cost of a small increase in path length, the prioritized approaches scale well with more number of robots. By choosing a proper prioritization methodology, the problems accommodating as many as 24 robots in the confined environments can be solved in a simplified manner compared with other coordination methods [27]. The work by [28] proposes a novel dynamic priority strategy for the multi-mobile robot path planning problem. In terms of the number of deadlocks, the dynamic priority-based path planning strategy proposed efficiently solved the coordination problem when compared with the static approach. However, in the present SwarmItFIX coordination problem, the task planning module completely avoids the possibility of deadlocks. Also, at any given point of time, one robot alone advances to the new location; whereas, the other robots hold the position while supporting the sheet metal. Based on these reasons, a static priority-based approach has been selected for the multi-robot path planning of SwarmItFIX robots.

Problem definition

The task is to perform the machining (milling) on the giant sheet metal, where the robots perform the machining and fixturing operations. A ceiling-mounted 6-axis serial manipulator with the milling tool as an end effector performs the machining operation, and the novel SwarmItFIX robots are utilized as robotic (RFA) fixtures. For achieving high accuracy in the machining process, the SwarmItFIX robots have to provide support at the appropriate places where the serial manipulator is currently performing the machining operation. To achieve the task mentioned above, two different coordination problems need to be addressed. The first one is the heterogeneous coordination between the ceiling-mounted serial manipulator robot and the group of SwarmItFIX robots. Second is the homogenous coordination among the multiple SwarmItFIX robots. These two coordination problems are addressed in this work by further dividing them into three different planning problems and presented in the next section.

Problem formulation

The coordination between COMAU NS16 serial manipulator robot and SwarmItFIX robots is achieved by solving three subproblems as given in the planner (Fig. 5). The three subproblems include:

- Trajectory planning of COMAU NS16.
- Multi-robot task planning of SwarmItFIX robots.
- Multi-robot path planning of SwarmItFIX robots.

Trajectory planning of COMAU NS16

In this work, the COMAU NS16 serial manipulator (Fig. 1b) is deployed to perform the sheet milling operation with milling tool as its end effector. The technical specifications of COMAU NS16 and its kinematic DH parameters are presented in Tables 1 [29] and 2, respectively. The corner of the workbench is considered the origin position (x_0, y_0, z_0) for all robots to reduce complexity. The total transformation between the base of serial manipulator and the TCP can be represented as homogenous transformation matrix given in Eq. (1). While developing the DH based inverse kinematics model of the ceiling-mounted serial manipulator robot, the base (${}^{sm}x_b, {}^{sm}y_b, {}^{sm}z_b$) of the robot is considered as its home position. The frame assignment of COMAU NS16 for the DH convention is shown in Fig. 4. The serial manipulator's trajectory is planned by considering its TCP velocity as constant, as the machining is considered to be performed at a constant feed rate. As the tool path is pre-defined, the trajectory planning is performed in the cartesian space. Initially, the vertex coordinates of the machining trajectory are fed into the trajectory planning model, then the whole trajectory is split into various line segments and each segment is again divided into numerous waypoints. Finally, required angles (${}^{sm}\theta_{k,w}$) of all the manipulator joints for all the waypoints are determined with the help of an optimization-based inverse kinematics model given in Eqs. (2–5). Where, Eq. (2) is the objective function, constraints (3) and (4) limits the angles of all the joints of serial manipulator. Equation (3) limits all joints angles ${}^{sm}\theta_k$, except ${}^{sm}\theta_1$. Where, Eq. (4) is a collision

Table 1 Joint limits of COMAU NS16

Joint	${}^{sm}\theta_k^+$ (°)	${}^{sm}\theta_k^-$ (°)
1	180	–180
2	155	–60
3	110	–170
4	2700	–2700
5	120	–120
6	2700	–2700

Table 2 DH table

Link (k)	${}^{sm}\theta_k$ (°)	${}^{sm}d_k$ (mm)	${}^{sm}a_k$ (mm)	${}^{sm}\alpha_k$ (°)
0	θ_0	–600	300	–90
1	θ_1-90	0	–700	0
2	θ_2	0	–185	–90
3	θ_3	624	0	90
4	θ_4	0	0	–90
5	θ_5	255	0	0
6 (tool)	–	0	–120	0

avoidance constraint that has been imposed exclusively for ${}^{sm}\theta_1$. Finally, the joint angles between two-way points (${}^{sm}\theta_{k,w-1}$ and ${}^{sm}\theta_{k,w}$) are interpolated in the joint space:

$${}^0T_7 = {}^0T_1 \times {}^1T_2 \times {}^2T_3 \times {}^3T_4 \times {}^4T_5 \times {}^5T_6 \times {}^6T_7$$

$$= \begin{bmatrix} n_x & o_x & a_x & {}^{sm}x_b + p_x \\ n_y & o_y & a_y & {}^{sm}y_b + p_y \\ n_z & o_z & a_z & {}^{sm}z_b + p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Objective function:

$$\text{Min} \left[{}^{sm}\theta_{k,w} - f \begin{pmatrix} {}^{sm}x_w \\ {}^{sm}y_w \\ {}^{sm}z_w \end{pmatrix} \right] \quad (2)$$

Constraints:

$${}^{sm}\theta_k^- \leq {}^{sm}\theta_{k,w} \leq {}^{sm}\theta_k^+; \text{ where, } k = 0, 2, 3, 4, 5, \quad (3)$$

$$0 \leq {}^{sm}\theta_{1,w} \leq {}^{sm}\theta_1^+, \quad (4)$$

Fig. 4 Frame assignment and work volume of COMAU NS16

Fig. 5 Hybrid coordination planner

$${}^{sm}\theta_{k,w-1} - {}^{sm}\theta_{k,w} \leq \eta. \quad (5)$$

Multi-robot task planning of SwarmItFIX robots

This module takes care of heterogeneous and homogenous coordination among the robots by computing the support locations and appropriate support duration and allocating them to respective SwarmItFIX robot. Initially, using ϕ_n , the corner support locations will be identified. Then, the

required number of support locations is calculated based on the respective segment length. Finally, the support locations are identified using the Monte Carlo-based head task assignment by considering the geometrical and FEM constraints. Once $({}^{head}x_{m,n}, {}^{head}y_{m,n}, {}^{head}z_{m,n})$ are identified, the time plan module computes $t_{m,n}$ for all the support locations as under:

$$t_{m,n} = \frac{\sqrt{\{ {}^{head}x_{m,n} + {}^{ch}D_m[\cos(90 - \phi_n)] - {}^{vert}x_n \}^2 + \{ {}^{head}y_{m,n} + {}^{ch}D_m[\sin(90 - \phi_n)] - {}^{vert}y_n \}^2}}{tool_v}. \quad (6)$$

As already mentioned, the milling process is assumed to be performed at a constant feed rate for better machining quality and hence the time plan computation for the serial manipulator becomes straightforward.

Task allocation

The total number of robots deployed is decided based on the size of the machining trajectory and workbench. Once the support locations and orientations ($^{head}x_{m,n}$, $^{head}y_{m,n}$, $^{head}z_{m,n}$, ϕ_n) are calculated, Task_m as presented in Eq. (7) will be allocated to a swarmitfix robot R_q :

$$\text{Task}_m = [\text{head}x_{m,n}, \text{head}y_{m,n}, \text{head}z_{m,n}, \phi_n, \text{baseGpos}_m, t_{m,n}]. \quad (7)$$

Initially, the Task_m is categorized into T_{odd} and T_{even} . If the total number of robots deployed is 2, then tasks T_{odd} and T_{even} are allocated to robot 1 (R_1) and robot 2 (R_2), respectively. If the total robots deployed is 3, then T_{even} is allocated to R_2 and T_{odd} is shared among R_1 and R_3 . If the mutual docking pins are present between two consecutive baseGpos_m , then task will be allocated to the current robot itself. Otherwise, the new robot will be deployed. If the total robots deployed is 4, T_{odd} is shared among R_1 and R_3 , and T_{even} among R_2 and R_4 (see Algorithm 1).

Algorithm 1 SwarmItFIX task allocation algorithm

```

If  $q=2$ 
     $T_{\text{odd}} \rightarrow R_1$ , and  $T_{\text{even}} \rightarrow R_2$ 
If  $q=3$ 
     $T_{\text{even}} \rightarrow R_2$ 
    If  $T_{\text{odd}}(\text{baseGpos}_m) \neq T_{\text{odd}}(\text{baseGpos}_{m-1})$ 
        If no_mutual_pins [ $T_{\text{odd}}(\text{baseGpos}_m)$ ,  $T_{\text{odd}}(\text{baseGpos}_{m-1})$ ]
             $T_{\text{odd}} \rightarrow R_3$ 
        Else
             $T_{\text{odd}} \rightarrow R_1$ 
If  $q=4$ 
    If  $T_{\text{odd}}(\text{baseGpos}_m) \neq T_{\text{odd}}(\text{baseGpos}_{m-1})$ 
        If no_mutual_pins [ $T_{\text{odd}}(\text{baseGpos}_m)$ ,  $T_{\text{odd}}(\text{baseGpos}_{m-1})$ ]
             $T_{\text{odd}} \rightarrow R_3$ 
        Else
             $T_{\text{odd}} \rightarrow R_1$ 
    If  $T_{\text{even}}(\text{baseGpos}_m) \neq T_{\text{even}}(\text{baseGpos}_{m-1})$ 
        If no_mutual_pins [ $T_{\text{even}}(\text{baseGpos}_m)$ ,  $T_{\text{even}}(\text{baseGpos}_{m-1})$ ]
             $T_{\text{even}} \rightarrow R_4$ 
        Else
             $T_{\text{even}} \rightarrow R_2$ 

```

Multi robot path planning of SwarmItFIX robots

The SARSA TD presented in this work is a model-free reinforcement learning algorithm that computes collision-free action sequence of the base agent alone. In addition, the five step sequence ensures collision free locomotion of the

complete SwarmItFIX robot based on the data received from head, PKM, and base planning submodules (Fig. 5). The five-step sequence for the locomotion of SwarmItFIX robot is explained below.

- Step 1: Retraction of head from previous support location ($^{head}x_{m-1,n}$, $^{head}y_{m-1,n}$, $^{head}z_{m-1,n}$).
- Step 2: PKM agent movement to keep the head in $^{pkm}\theta_{j,\text{park}}$.
- Step 3: Locomotion sequence of the base agent ($\text{baseGpos}_{m-1} \rightarrow \text{baseGpos}_m$) and the respective counter-rotation of PKM.
- Step 4: The PKM agent movement keeps the head at a normal approach away from $^{pkm}\theta_{j,m}$.
- Step 5: Head agent attains ($^{head}x_{m,n}$, $^{head}y_{m,n}$, $^{head}z_{m,n}$) and $^{head}\theta_{m,n}$.

The head agent's positioning in the target coordinate ($^{head}x_{m,n}$, $^{head}y_{m,n}$, $^{head}z_{m,n}$) is achieved by the inverse position kinematics model of the PKM agent [30]; whereas, the $^{head}\theta_{m,n}$ is wholly related to the ϕ_n . When ($^{head}x_{m,n}$, $^{head}y_{m,n}$, $^{head}z_{m,n}$) is located outside the work envelope of PKM agent, the robot traverses to the new location baseGpos_m with the help of the base agent's SaD locomotion. As more than one robot is being deployed and the work environment becomes dynamic, the robots currently providing the support to the sheet metal become obstacles ($^{\text{obs}}\text{Task}_m$) to the traversing robot. Hence, in order to compute the collision-free locomotion sequence of the SwarmItFIX robots, a novel decentralized type prioritized offline multi-agent motion planning technique (see Algorithm 2) is utilized in this work with SARSA TD as the base algorithm.

MDP modeling

When the SwarmItFIX robot is assigned with Task_m by the task allocation module, the computational environment (Fig. 6) which replicates the real-time environment is created accordingly. Then the prioritized SARSA algorithm is utilized to compute the collision-free path of base agent.

Various RL parameters of the SwarmItFIX base agent are given below.

State-space (S): The states ($s \in S$) replicates the different positions of the workbench. The number of states in S is calculated using Euler's formula (8). Current positions of other robots will also be noted, and the algorithm considers those positions as $^{\text{obs}}\text{Task}_m$ for the $R_q\text{Bseq}_m$ of the traversing robot R_q . The states which share mutual pins with obstacle states are also considered as the $^{\text{obs}}\text{Task}_m$ (Fig. 6):

$$N_{\text{state}} = 1 - N_{\text{pins}} + N_{\text{edge}}. \quad (8)$$

Fig. 6 MDP environment—SwarmItFIX robot

The workbench of the SwarmItFIX consist of 129 edges (N_{edge}) and 52 docking pins (N_{pins}). Using Eq. (8), the states (N_{states}) is calculated as 78 as shown in Fig. 6.

Action space (A): The actions ($a \in A$) of the computational agent replicates the physical locomotion of the base agent of the SwarmItFIX. The action space consists of nine different actions, the details of which are given in appendix A.

Reward matrix $R(s, a)$: The reward value for any particular state s is calculated based on the Euclidean distance between $^{\text{base}}\text{Cpos}_q$ and $^{\text{base}}\text{Gpos}_m$ (9). Negative rewards will be given to the $^{\text{obs}}\text{Task}_m$:

$$R(s, a) = \begin{cases} \left(1 - \left\| ^{\text{base}}\text{Cpos}_q - ^{\text{base}}\text{Gpos}_m \right\| \right), & \text{if } ^{\text{base}}\text{Cpos}_q = ^{\text{base}}\text{Gpos}_m \\ -100, & \text{elif } ^{\text{base}}\text{Cpos}_q = ^{\text{obs}}\text{Task}_m \\ -\left\| ^{\text{base}}\text{Cpos}_q - ^{\text{base}}\text{Gpos}_m \right\|, & \text{otherwise} \end{cases} \quad (9)$$

State-action matrix ($Q(s, a)$): The utility values for all the actions at all states are calculated and updated in the Q-table $Q(s, a)$. The size of $Q(s, a)$ is 78 states $\times$ 9 actions.

Implementation

The trajectory details ($^{\text{sm}}\text{Traj}$) computed by trajectory planning module of the serial manipulator module and locomotion sequence ($^{\text{Rq}}\text{seq}$) obtained by the multi-robot path planning module of SwarmItFIX is finally communicated to the implementation module. In the implementation

module, $^{sm}Traj$ is converted into PDL2 program and fed to the COMAU NS16 controller. Similarly, the ^{Rq}seq is fed to the SwarmItFIX controller using MRROC++ interface.

Algorithm 2 Prioritized SARSA

```

Initialize:  $Task_m, {}^{rob}Task_m, {}^{obs}Task_m, {}^{base}Cpos_q = {}^{base}Hpos_q$ 
For  $n$  in range ( $m$ )
     $robot = {}^{rob}Task_m$ 
     $obstacles = {}^{obs}Task_m$ 
     $Start = {}^{base}Cpos_q$ 
     $goal = {}^{base}Gpos_m$ 
    • Input:
        ◦ Step size  $\alpha \in (0, 1]$ , small  $\epsilon > 0$ 
        ◦  $\pi(s) \in A$  (arbitrarily), for all  $s \in S$ 
        ◦  $Q(s, a) \in \mathbb{R}$  (arbitrarily), for all  $s \in S, a \in A(s)$ 
    • Output: Optimal policy  $\pi^*$ , updates  $Q(s, a)$ 
    • Loop for each episode:
        ◦ Initialize  $s \in S$ 
        ◦ Choose  $a \in A(s)$  using an epsilon greedy policy.
        ◦ Loop for each step of the episode
            ▪ Take action  $a$ , observe  $R(s, a), s'$ 
            ▪ Choose  $a' \in A(s')$  using an epsilon greedy policy
            ▪  $Q(s, a) \leftarrow Q(s, a) + \alpha[R(s, a) + \gamma Q(s', a') - Q(s, a)]$ 
            ▪  $s \leftarrow s', a \leftarrow a'$ 
        ◦ Until  $s$  is a terminal state
    • Loop for all states
        ◦  $\pi(s) = \underset{a \in A(s)}{\operatorname{argmax}} Q(s, a)$ 
    • Return  $\pi^*$ 
 ${}^{Rq}Bseq_m = Path({}^{Rq}\pi_m^*, Start)$ 
 ${}^{base}Cpos_q = {}^{base}Gpos_m$ 

```

Results and discussions

Results of serial manipulator

The configuration of system utilized to perform this multi-robot coordination planning computation includes Intel Xeon CPU E3-1225v6 Processor 3.30 GHz, Windows 10, 64-bit OS with 16.0 GB RAM. The numerical inverse kinematics-based trajectory planning of the COMAU NS16 serial manipulator is performed using the Robotics system toolbox of Matlab. Task planning and the homogenous prioritized multi-robot planning of the SwarmItFIX (RFA) robots are performed using python 3.7 idle interface, and the results are presented in this section.

Trajectory planning of serial manipulator

The following parameters are assumed for the purpose of trajectory planning of the serial manipulator. The joint angles of the manipulator at the ${}^{sm}\theta_{k,home}$ are kept as $(180^\circ, 0^\circ, -90^\circ, 0^\circ, 0^\circ, 0^\circ)$, so that the first vertex coordinate (${}^{vert}x_1, {}^{vert}y_1, {}^{vert}z_1$) is attained with the shorter transformation and ${}^{tool}v$ is considered as 5 mm/s [16]. The front left corner of the workbench is considered the origin (x_0, y_0, z_0) , and the rest of the coordinates are assumed by keeping (x_0, y_0, z_0) as reference. Table 3 shows various coordinate frame values.

The plot in Fig. 7a show the details of the position, velocity, and acceleration of TCP for the complete machining trajectory. Angles of all joints of the COMAU NS16 at all the waypoints computed by the numerical inverse kinematics model are detailed in Fig. 7b. The positional error (Fig. 7c) of the TCP frame obtained during simulation is observed to be very low throughout the trajectory, which portrays the efficiency of the inverse kinematics-based trajectory planning model. Figure 7d show the iterations consumed to compute the joint angles at each waypoint. The data conveys the computational efficiency of the numerical inverse kinematics technique deployed. The peak noticed at the beginning denotes a higher number of iterations consumed for calculating the angles of joints 2, 5, and 6. This peak is due to higher variation between joint angles of neighboring waypoints (see Fig. 7b). The bumps in the joint angles happen because the reach at those waypoints is minimal. It can also be observed that no sudden variations are present throughout the trajectory. Hence, the trajectory obtained is very smooth. Figure 7e represent the home position (${}^{sm}\theta_{k,home}$) of the serial manipulator. Figure 7f–h are related to joint variables when time $t=2, 350$, and 700 s of the machining process, respectively.

Multi-robot task planning of SwarmItFIX robots

Results obtained from the SwarmItFIX task planning module are given in Table 4. The Monte Carlo-based MDP algorithm identifies a total of 30 different support locations

Table 3 Coordinates of vertices of polygonal machining trajectory

S. No.	Frame	x (m)	y (m)	z (m)
1	Origin {0}	0	0	0
2	COMAU base{b}	1.000	1.085	2.661
3	Vertex 1	0.914	1.016	1.561
4	Vertex 2	1.1615	1.5453	1.561
5	Vertex 3	1.7432	1.5961	1.561
6	Vertex 4	2.0785	1.1176	1.561
7	Vertex 5	1.8313	0.5883	1.561
8	Vertex 6	1.2497	0.5374	1.561

Fig. 7 Results of trajectory planning of serial manipulator

Table 4 Results of task planning module

Support position	R_q	Support coordinates		$t_{m,n}$ (s)	Support position	R_q	Support coordinates		$t_{m,n}$ (s)
		$\text{head}_{x_{m,n}}$ (m)	$\text{head}_{y_{m,n}}$ (m)				$\text{head}_{x_{m,n}}$ (m)	$\text{head}_{y_{m,n}}$ (m)	
1	R_1	0.979	1.061	18.9	2	R_2	1.015	1.158	35.1
3		1.074	1.247	58.3	4		1.111	1.364	80.3
5		1.168	1.467	105.8	6		1.233	1.512	129.5
7		1.335	1.529	151.5	8		1.442	1.522	171.1
9		1.562	1.549	196.7	10		1.679	1.551	218.6
11		1.750	1.518	248.3	12		1.816	1.438	268.1
13		1.864	1.342	285.9	14		1.947	1.251	313.3
15		2.007	1.151	333.1	16		2.014	1.073	370.0
17		1.978	0.975	385.1	18		1.918	0.886	404.9
19		1.881	0.769	430.2	20		1.825	0.667	451.7
21		1.760	0.622	483.3	22		1.657	0.604	502.3
23		1.551	0.611	525.7	24		1.431	0.585	547.4
25		1.314	0.583	572.5	26		1.243	0.616	595.9
27		1.177	0.696	619.3	28		1.129	0.792	644.3
29		1.046	0.882	664.5	30		0.986	0.983	691.5

($\text{head}_{x_{m,n}}$, $\text{head}_{y_{m,n}}$, $\text{head}_{z_{m,n}}$) throughout the trajectory, and the time duration also estimated using Eq. (5). Finally, the tasks obtained are allocated for both the SwarmItFIX robots based on the procedure established in algorithm 1. Figure 10 proves that the data computed by the task planning module makes both the robots extend the support at the appropriate coordinates at the right time in-line with the machining tool trajectory.

Homogenous coordination among SwarmItFIX

The positions C66 and C1 are considered as the home positions baseHpos_q of base agent of SwarmItFIX robot 1 (R_1) and robot 2 (R_2), respectively. The base agent abides by the sub-optimal policies (π^*) computed by the prioritized multi-robot path planning module and obtains the locomotion sequence for attaining the goal positions. Finally, both the robots follow the five-step sequence and attain the goal positions consecutively. Figure 8a represents the sub-optimal policy ($R^1\pi_3^*$) of the base agent of R_1 . Position C7 is the goal state, and positions C32–C34, C45–C47, and C58–C60 have to be avoided as these positions were occupied by R_2 . Sub-optimal policy $R^1\pi_3^*$ guides R_1 to attain the goal position without colliding with R_2 by providing the optimal sequence of actions. In all states, the action with the highest utility value is considered the optimal action. It can be observed from Fig. 8b that action (a_1) completes all iterations with the highest utility value among other actions at C20. Thus, $R^1\pi_3^*$ suggests action (a_1) as the optimal action for position C20. Similarly, Fig. 8c represent the sub-optimal policy for the base agent of R_2 ($R^2\pi_1^*$) for attaining the goal position C7. The optimal action convergence for the state C78 is shown in Fig. 8d. One can also note that the utility values are null

for all actions at goal positions as well as obstacle positions. If looked closely, the optimal action in Fig. 8b converges within few iterations, which is very much faster compared with the convergence of optimal action of the second case (Fig. 8d). This is because the distance between baseCpos_q and baseGpos_m is comparatively less in the first case. Figure 9 depicts the transformation of π into π^* shown in Fig. 8a and c. It can be observed that during the initial phase of iterations, the computational agent performs actions suggested by random policy (π) and requires more than 500 steps for attaining the goal position. When the iteration progresses, the π is slowly becomes optimistic and then the computational agent performs a lesser number of actions to reach the goal position. The sequence of locomotion of the SwarmItFIX robot and the time plan of all the three robots are detailed in Figs. 10 and 11, respectively. The complete details of the sequence of agents of SwarmItFIX robot are presented in appendix B. The horizontal bars (Fig. 11) show the duration of support extended by the SwarmItFIX robot at 30 different support locations, and the line of interpolation shows the movement of the serial manipulator robot. This result confirms on the optimal coordination and time synchronization amongst all the three robots.

The SwarmItFIX robots extend the support in appropriate locations at the right time when the machining is being performed. In all the planned locations, the supports are provided in line with the movement of the TCP. This synchronization ultimately portrays the efficiency of the proposed planner. In the hexagonal tool trajectory, whenever the robot switches between two segments, a sudden jerk in the tool path is observed (near positions 11, 16, 21 in Fig. 11). However, at the 16th position, the robot achieves the comparatively longer jerk in the tool path (between 350

Fig. 8 Sub-optimal policies and utility values of actions space of the state

Fig. 9 Sub-optimal policy convergence

and 400 s). This jerk is due to the fact that, the robot reaches the farther points in the x -axis of the tool trajectory (higher reach) during the specified duration. The same can be seen in Fig. 7a also. As mentioned in the introductory section, if

the reach is maximum the accuracy and repeatability will be of minimum.

Fig. 10 SwarmItFIX agents placement

Fig. 11 Coordination and time synchronization of COMAU NS16 and SwarmItFIX

Conclusion and future work

The proposed hierarchical-based hybrid type decentralized offline planner successfully coordinate all three robots in sheet metal milling process.

The numerical optimization-based inverse kinematics has been utilized for computing the joint angles for the COMAU NS16 manipulator by considering various constraints. To

ensure the collision-free trajectory of the serial manipulator robot with the workpiece, a constraint (4) on ${}^{sm}\theta_{1,w}$, which restricts the negative set of angles, has been imposed. Also, in order to generate a smoother trajectory (without abrupt changes in joint angles between two-way points), a constraint (5) that ensures the shorter translation is also imposed. The same can be verified from Fig. 6b. The position, velocity, and acceleration of TCP for complete trajectory have been

computed without any swift changes in joint angles and collision.

The proposed novel prioritized multi-robot path planning approach successfully computes the sub-optimal policies for both the robots for all goal positions. The SARSA algorithm consumes 616.2 s for computation of sub-optimal policies of base agent of the SwarmItFIX robot 1. Whereas, for base agent of the SwarmItFIX robot 2, it took only 196.4 s. The reason being, first, the R_2 utilizes only five base positions repeatedly for the total of 15 different tasks. But the robot 1 uses nine base positions for the same number of tasks (Fig. 10). Second, R_1 attains all 15 different base goal positions by performing 22 number of SaD steps. But the R_2 only performs 10 SaD steps. From the above data, it can be clearly understood the R_2 traverses within the small area of the workbench, whereas, the R_1 covers the larger area.

The algorithm 1 is able to perform task allocation up to four number of SwarmItFIX robots. But this work only considers only two SwarmItFIX robots. However, this number can be increased depends on the availability of the more robots in the experimental setup. For the computation of sub-optimal policies of the base agent, a constant ceiling number of iterations are performed. This constant number is because the utility value convergence duration is not the same for all states. It varies depends on its distance from the goal state. However, the computation can be further optimized by developing a mathematical relation that returns the optimal number of iterations using this distance between the current position and goal position and as a factor.

For ceiling-mounted COMAU NS16 manipulator is concerned, this work only portrays its trajectory planning only. Investigation on analysis of machining efficiency, machining quality, and stiffness analysis of all joints of the COMAU NS16 manipulator in the ceiling-mounted (inverted) position.

Appendix A: actions, locomotion mapping

Sl. No.	Action	Base locomotion						Next position
		If the current state is even			If the current state is odd			
		Leg	direction	Angle (°)	Leg	direction	Angle (°)	
1	a_0	No locomotion and stay in the current position						$baseC_{pos_q}$
2	a_1	1	CCW	60	2	CCW	180	$baseC_{-pos_q + 13}$
3	a_2	3	CW	120	2	CCW	120	$baseC_{-pos_q + 14}$
4	a_3	2	CW	60	2	CCW	60	$baseC_{-pos_q + 1}$
5	a_4	2	CW	120	3	CCW	120	$baseC_{-pos_q - 12}$
6	a_5	2	CCW	180	1	CW	60	$baseC_{-pos_q - 13}$
7	a_6	2	CCW	120	1	CW	120	$baseC_{-pos_q - 14}$
8	a_7	1	CW	60	1	CCW	60	$baseC_{-pos_q - 1}$
9	a_8	1	CCW	120	2	CW	120	$baseC_{-pos_q + 12}$

CW clockwise, CCW counter clockwise

Appendix B: motion sequence of the SwarmItFIX robot

The locomotion sequence of SwarmItFIX robots for the hexagonal trajectory is as follows

Step	Support location m	Robot R_q	Base agent details			Head agent details		
			$baseC_{pos_q}$	Corresponding positions (row, col)	Active docking pins	$head_{x_{m,n}}$ (m)	$head_{y_{m,n}}$ (m)	$head_{\theta_{m,n}}$ ($^{\circ}$)
1	–	2	C2	0,1	(P39, P46, P47)	Parking/rest position		
2	–		C3	0,2	(P39, P40, P47)	Parking/rest position		
3	–		C4	0,3	(P40, P47, P48)	Parking/rest position		
4	–		C18	1,4	(P33, P40, P41)	Parking/rest position		
5	–		C32	2,5	(P26, P33, P34)	Parking/rest position		
6	2		C46	3,6	(P19, P26, P27)	0.979	1.061	64.97
7	3	1	C29	2,2	(P24, P31, P32)	1.015	1.158	64.97
8	4	2	C46	3,6	(P19, P26, P27)	1.074	1.247	64.97
9	5	1	C29	2,2	(P24, P31, P32)	1.111	1.364	64.97
10	6	2	C46	3,6	(P19, P26, P27)	1.168	1.467	64.97
11	–	1	C43	3,3	(P17, P18, P25)	Parking/rest position		
12	–		C31	2,4	(P25, P26, P33)	Parking/rest position		
13	–		C19	1,5	(P33, P34, P41)	Parking/rest position		
14	7		C7	0,6	(P41, P42, P49)	1.233	1.512	4.99
15	8	2	C46	3,6	(P19, P26, P27)	1.336	1.529	4.99
16	9	1	C7	0,6	(P41, P42, P49)	1.442	1.522	4.99
17	10	2	C46	3,6	(P19, P26, P27)	1.562	1.549	4.99
18	–	1	C8	0,7	(P42, P49, P50)	Parking/rest position		
19	–		C9	0,8	(P42, P43, P50)	Parking/rest position		
20	–		C23	1,9	(P35, P36, P43)	Parking/rest position		
21	11		C11	0,10	(P43, P44, P51)	1.679	1.551	4.99
22	12	2	C34	2,7	(P27, P34, P35)	1.750	1.518	– 54.98
23	13	1	C11	0,10	(P43, P44, P51)	1.816	1.437	– 54.98
24	14	2	C34	2,7	(P27, P34, P35)	1.864	1.864	– 54.98
25	–	1	C25	1,11	(P36, P37, P44)	Parking/rest position		
26	15		C39	2,12	(P29, P30, P37)	1.947	1.251	– 54.98
27	16	2	C34	2,7	(P27, P34, P35)	2.007	1.151	– 54.98
28	17	1	C51	3,11	(P21, P22, P29)	2.014	1.072	244.97
29	18	2	C34	2,7	(P27, P34, P35)	1.978	0.975	244.97
30	19	1	C51	3,11	(P21, P22, P29)	1.918	0.886	244.97
31	20	2	C34	2,7	(P27, P34, P35)	1.881	0.769	244.97
32	–	1	C63	4,10	(P13, P14, P21)	Parking/rest position		
33	21		C75	5,9	(P5, P6, P13)	1.824	0.667	244.97
34	22	2	C48	3,8	(P20, P27, P28)	1.760	0.621	184.99
35	–	1	C74	5,8	(P5, P12, P13)	Parking/rest position		
36	23		C73	5,7	(P4, P5, P12)	1.657	0.604	184.99
37	24	2	C48	3,8	(P20, P27, P28)	1.551	0.611	184.99
38	–	1	C72	5,6	(P4, P11, P12)	Parking/rest position		
39	25		C71	5,5	(P3, P4, P11)	1.431	0.584	184.99
40	26	2	C47	3,7	(P19, P20, P27)	1.314	0.583	184.99
41	27	1	C71	5,5	(P3, P4, P11)	1.243	0.616	125.02
42	28	2	C46	3,6	(P19, P26, P27)	1.177	0.696	125.02
43	–	1	C57	4,4	(P10, P11, P18)	Parking/rest position		
44	29		C56	4,3	(P10, P17, P18)	1.129	0.792	125.02
45	30	2	C46	3,6	(P19, P26, P27)	1.046	0.882	125.02
46	–	1	C54	4,1	(P9, P16, P17)	Parking/rest position		
47	–		C41	3,1	(P16, P17, P24)	Parking/rest position		
48	1		C29	2,2	(P24, P31, P32)	0.986	0.983	125.02

– Indicates the clockwise direction of head agent

Acknowledgements This work is an outcome of the sponsored project (Project No.: MEC/DIMEUG-ITALY/2015-16/MSK/008/02) with financial Grant of the University of Genoa, Italy. The support being received from Prof. Matteo Zoppi, Prof. Rezia Molino, and Dr. Keerthi Sagar of PMAR Research Group @ The University of Genoa, Italy is highly acknowledged.

Code availability The codes are available with the authors, and can be shared upon request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Molino RM, Zoppi M, Zlatanov D (2013) Bench and method for the support and manufacturing of parts with complex geometry, US patent, US8495811B2
- Veeramani S, Muthuswamy S, Sagar K, Zoppi M (2020) Artificial intelligence planners for multi-head path planning of SwarmIt-FIX agents. *J Intell Manuf* 31:815–832. <https://doi.org/10.1007/s10845-019-01479-8>
- Veeramani S, Muthuswamy S, Sagar K, Zoppi M (2019) Multi-head path planning of SwarmIt-FIX agents : a Markov decision process approach. In: Proc. the 15th IFToMM world congress, Krakow, Poland. Springer International Publishing, pp 2237–2247
- Veeramani S, Muthuswamy S (2020) Reinforcement learning based path planning of the mobile agents with constrained locomotion for the material handling applications. In: 2020 IEEE 4th conference on information & communication technology (CICT). IEEE, pp 1–5
- Zieliński C, Kasprzak W, Kornuta T et al (2013) Control and programming of a multi-robot-based reconfigurable fixture. *Ind Rob* 40:329–336. <https://doi.org/10.1108/01439911311320831>
- Olabi A, Béarée R, Gibaru O, Damak M (2010) Feedrate planning for machining with industrial six-axis robots. *Control Eng Pract* 18:471–482. <https://doi.org/10.1016/j.conengprac.2010.01.004>
- Takeuchi Y, Asakawa N, Ge D (1993) Automation of polishing work by an industrial robot: system of polishing robot. *JSME Int J Ser C Dyn Control Robot Des Manuf* 36:556–561. <https://doi.org/10.1299/jsmec1993.36.556>
- Chen Y, Dong F (2013) Robot machining: recent development and future research issues. *Int J Adv Manuf Technol* 66:1489–1497. <https://doi.org/10.1007/s00170-012-4433-4>
- Dumas C, Caro S, Garnier S, Furet B (2011) Joint stiffness identification of six-revolute industrial serial robots. *Robot Comput Integr Manuf* 27:881–888. <https://doi.org/10.1016/j.rcim.2011.02.003>
- Chen C, Peng F, Yan R et al (2019) Stiffness performance index based posture and feed orientation optimization in robotic milling process. *Robot Comput Integr Manuf* 55:29–40. <https://doi.org/10.1016/j.rcim.2018.07.003>
- Xiong G, Ding Y, Zhu LM (2019) Stiffness-based pose optimization of an industrial robot for five-axis milling. *Robot Comput Integr Manuf* 55:19–28. <https://doi.org/10.1016/j.rcim.2018.07.001>
- Busoniu L, Babuska R, De Schutter B (2008) A comprehensive survey of multiagent reinforcement learning. *IEEE Trans Syst Man Cybern Part C Appl Rev* 38:156–172. <https://doi.org/10.1109/TSMCC.2007.913919>
- Sutton RS, Barto AG (2018) Reinforcement learning, an introduction, 2nd edn. The MIT Press, Cambridge
- Yan Z, Jouandeau N, Cherif AA (2013) A survey and analysis of multi-robot coordination. *Int J Adv Robot Syst*. <https://doi.org/10.5772/57313>
- Kasprzak W, Zlatanov D, Szykiewicz W, Zielińska T (2013) Task planning for cooperating self-reconfigurable mobile fixtures. *Int J Adv Manuf Technol* 69:2555–2568. <https://doi.org/10.1007/s00170-013-5185-5>
- Kasprzak W, Szykiewicz W, Zlatanov D, Zielińska T (2014) A hierarchical CSP search for path planning of cooperating self-reconfigurable mobile fixtures. *Eng Appl Artif Intell* 34:85–98. <https://doi.org/10.1016/j.engappai.2014.05.013>
- Thejas SS, Chittawadigi RG, Prasanth S et al (2019) Design analysis of an industrial robot under development. *ACM Int Conf Proc Ser*. <https://doi.org/10.1145/3352593.3352612>
- Zhong J, Wang T, Cheng L (2021) Collision-free path planning for welding manipulator via hybrid algorithm of deep reinforcement learning and inverse kinematics. *Complex Intell Syst*. <https://doi.org/10.1007/s40747-021-00366-1>
- Starke S, Hendrich N, Zhang J (2018) Memetic evolution for generic full-body inverse kinematics in robotics and animation. *IEEE Trans Evol Comput* 23(3):406–420. <https://doi.org/10.1109/TEVC.2018.2867601>
- Zolfpour-Arokhlo M, Selamat A, Mohd Hashim SZ, Afkhami H (2014) Modeling of route planning system based on Q value-based dynamic programming with multi-agent reinforcement learning algorithms. *Eng Appl Artif Intell* 29:163–177. <https://doi.org/10.1016/j.engappai.2014.01.001>
- Chouhan SS, Niyogi R (2017) DiMPP: a complete distributed algorithm for multi-agent path planning. *J Exp Theor Artif Intell* 29:1129–1148. <https://doi.org/10.1080/0952813X.2017.1310142>
- Arokiasami WA, Vadakkepat P, Tan KC, Srinivasan D (2016) Interoperable multi-agent framework for unmanned aerial/ground vehicles: towards robot autonomy. *Complex Intell Syst* 2:45–59. <https://doi.org/10.1007/s40747-016-0014-8>
- Chouhan SS, Niyogi R (2015) DMAPP: a distributed multi-agent path planning algorithm, pp 123–135. https://doi.org/10.1007/978-3-319-26350-2_11
- Okoso A, Otaki K, Nishi T (2019) Multi-agent path finding with priority for cooperative automated valet parking. In: 2019 IEEE Intell Transp Syst Conf ITSC 2019, pp 2135–2140. <https://doi.org/10.1109/ITSC.2019.8917112>
- Dewangan RK, Shukla A, Godfrey WW (2018) Prioritized multi robot path planning in presence of dynamic obstacles. In: 2017 Conf Inf Commun Technol CICT 2017 2018-April, pp 1–5. <https://doi.org/10.1109/INFOCOMTECH.2017.8340595>
- Dewangan RK, Shukla A, Godfrey WW (2017) Survey on prioritized multi robot path planning. In: 2017 IEEE Int Conf Smart Technol Manag Comput Commun Control Energy Mater ICSTM 2017 – Proc, pp 423–428. <https://doi.org/10.1109/ICSTM.2017.8089197>

27. Van Den Berg JP, Overmars MH (2005) Prioritized motion planning for multiple robots. In: 2005 IEEE/RSJ Int Conf Intell Robot Syst IROS, pp 430–435. <https://doi.org/10.1109/IROS.2005.1545306>
28. Liu S, Sun D, Zhu C (2014) A dynamic priority based path planning for cooperation of multiple mobile robots in formation forming. *Robot Comput Integr Manuf* 30:589–596. <https://doi.org/10.1016/j.rcim.2014.04.002>
29. Villagrossi E (2017) Robot dynamics modelling and control for machining applications. University of Brescia
30. Zoppi M, Zlatanov D, Molino R (2010) Kinematics analysis of the exechon tripod. In: Proc. ASME international design engineering technical conferences & computers and information in engineering conference IDETC/CIE, Montreal, Quebec, Canada. Proceedings of the ASME 2010, pp 1–10

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.