Article:
Canzini, E., Auledas-Noguera, M., Pope, S. et al. (2024) Decision making for multi-robot fixture planning using multi-agent reinforcement learning. IEEE Transactions on Automation Science and Engineering, 22. pp. 5578-5589. ISSN: 1545-5955

https://doi.org/10.1109/TASE.2024.3424677

Decision Making For Multi-Robot Fixture Planning Using Multi-Agent Reinforcement Learning

Ethan Canzini, *Member, IEEE*, Marc Auledas-Noguera, Simon Pope, Ashutosh Tiwari

***Abstract*—Within the realm of flexible manufacturing, fixture layout planning allows manufacturers to rapidly deploy optimal fixturing plans that can reduce surface deformation that leads to crack propagation in components during manufacturing tasks. The role of fixture layout planning has evolved from being performed by experienced engineers to computational methods due to the number of possible configurations for components. Current optimisation methods commonly fall into sub-optimal positions due to the existence of local optima, with data-driven machine learning techniques relying on costly to collect labelled training data. In this paper, we present a framework for multi-agent reinforcement learning with team decision theory to find optimal fixturing plans for manufacturing tasks. We demonstrate our approach on two representative aerospace components with complex geometries across a set of drilling tasks, illustrating the capabilities of our method; we will compare this against state of the art methods to showcase our method's improvement at finding optimal fixturing plans with 3 times the improvement in deformation control within tolerance bounds.**

***Note to Practitioners*—Fixture layout planning is one of the most fundamental manufacturing tasks that must be carried out before production cycles can begin, to ensure that deformation is within tolerances to avoid crack propagation and component damage. However, reliance on humans to generate these plans has led to sub-optimal solutions, leading to manufacturers incurring losses and wanting to explore analytical methods for fixture planning. In this vein, there may be the temptation to find a single solution that can be applied to all problems regardless of complexity. However, in the age of flexible manufacturing, the benefits of building tailored solutions within a framework - referred to as "freedom within a framework" - become more apparent. This paper outlines the framework for multi-agent reinforcement learning for fixture layout planning, and demonstrates the capabilities of this framework to outperform current state of the art methods with a simple algorithm through extensive experiments on representative aerospace components. We provide code implementations of our work on GitHub[1]**

***Index Terms*—Multi-Agent Systems; Reinforcement Learning; Aerospace Manufacturing; Fixture Planning; Robotic Fixturing**

The work of Ethan Canzini was supported by the EPSRC ICASE Award with Airbus UK and was co-sponsored by the University of Sheffield and Airbus UK Assembly Technologies. This work received funding from the UKRI EPSRC Made Smarter Innovation-Research Centre for Connected Factories (Grant EP/V062123/1) and the RAEng/Airbus Research Chairs & Senior Research Fellowships Scheme (Grant RCSRF1718/5/41). Ashutosh Tiwari is Airbus/RAEng Research Chair in Digitisation for Manufacturing at the University of Sheffield.

All authors are with the Department of Automatic Control & Systems Engineering, University of Sheffield, UK. Ethan Canzini is a Research Scientist with Airbus Robotics

Corresponding Author: ecanzini1@sheffield.ac.uk

[1]Multi-Agent Fixture Planner on GitHub

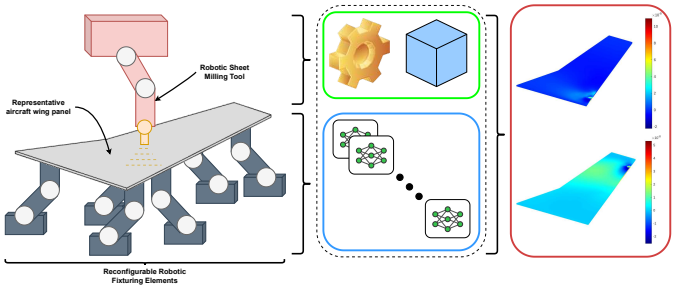


Fig. 1. A set of robotic flexible fixtures working with a serial manipulator for aerospace sheet metal milling where task and component information (GREEN) is used with a multi-agent reinforcement learning network (BLUE) allows reconfigurable fixtures to reduce deformation during tasks (RED)

I. Introduction

With the increased demand placed on producers to meet the needs of consumers, manufacturing practises and techniques have seen a revolution that signals the beginning of new paradigms in manufacturing. Dubbed *Industry 4.0*, this new industrial revolution has steered large scale industries to deploy technological trends such as autonomous robotics, artificial intelligence (AI) and digital simulation systems to their production benefit. However, this revolution has brought the realisation that manufacturers need to become flexible to meet the shifting demands of consumers. This flexibility has led to the rise of reconfigurable machines for industrial production so customer demands can be met efficiently.

An area that has seen a rise in interest is the use of intelligent fixtures for reconfigurable manufacturing purposes. Fixtures - referred to as fixturing/fixel elements interchangeably - are used within jig structures to constrain a component to complete manufacturing tasks such as drilling, joining or riveting [1]. These fixtures are used to ensure that there is no deformation that is outside of tolerance limits, to mitigate the possibility of large component deflections and the probability of internal cracks propagating through the component. In the past, these jigs were large static structures that constrained a component without allowing for any flexibility or reconfiguration. Recently, reconfigurable fixtures have seen an increase in research in two particular areas:

1) The physical design of fixturing elements such as clamps, pin arrays and robotic fixtures, such as those in figure 1
2) Design of the fixturing plan that determines where the fixturing elements are placed on the component

The design of physical fixtures, particularly robotic end effectors which have received the most interest in recent years,

remains a discussion for manufacturers as it is dependent on the type of component that is being constrained. However, the process of algorithmic fixture planning can be applied to robotics fixtures and static fixtures alike. The goal of fixture planning is to find the location of fixturing elements in such a way that they minimise any potential residual stress or deformation a component will experience when it undergoes a manufacturing task. This task is crucial to the long-term success of manufacturers to ensure their components retain their strength and capability. However, such a task requires the coordination of mechanical elements acting in parallel with material properties of components. These components are subjected to operational forces that can distort a component if not constrained correctly, leading to a non-trivial task for determining optimal plans. Historically, the design of such plans fell to experienced design engineers with years of experience, but research has also examined the use of optimisation-based approaches for determining the location of the elements. Despite these advances, fixture layout planning remains an open problem due to the nature of the drawbacks in optimisation methods [1].

As an alternative to optimisation methods, reinforcement learning has emerged as a useful method for finding optimal actions. Traditional methods have relied on a single agent, defined in the context of this problem as a single robotic fixture, operating in an environment and being able to interact individually. However, many real-world systems deal with multiple agents that have distributed decision making within a shared environment.

This paper will introduce a framework for multi-agent reinforcement learning (MARL) when applied to the fixture layout planning problem for multiple robotic fixturing elements working in a cooperative manner. The layout of the paper will be as follows: Section II will review current fixture planning methods and their limitations followed by an overview of multi-agent reinforcement learning. Section III will introduce the *Multi-Robot Fixture Planner* (MRFP) methdology and detail the process in which optimal fixture plans are computed. Section IV will evaluate the performance of the MRFP method on a case study for aerospace wing panel and spar drilling and compare the performance against state of the art methods for fixture planning. Finally, section VI will summarise the work presented in this paper and discuss any directions or further work that needs to be completed. Mathematical notation will follow conventions in [2] for fixture design, [3] for multi-agent reinforcement learning and [4] for team theory.

II. Related Work

This section encompasses a brief review and introduction to the work in this paper. Section II-A will provide an overview of fixture design and current approaches to planning. Section II-B will introduce multi-agent reinforcement learning (MARL) and provide examples of current state of the art methods. Section II-C outlines the gap in the research area.

A. Fixture Layout Planning

As mentioned in section I, fixture design for manufacturing processes is broken down into the design of physical fixtures and the grasping strategy for component constraining. Whilst physical fixture design has seen relevant innovation through the use of robotic fixturing elements, the design of fixture plans has remained a key challenge as this is a task primarily performed by operators with considerable experience in the field. The purpose of these plans is to ensure that a component is fully constrained before performing a task, commonly drilling or riveting [5] such that any experienced deformation is minimised.

Definition 1: Let $A \subseteq \mathcal{A}$ be a set of fixture locations chosen from the global set of positions $\mathcal{A}$. The locations of A during a manufacturing operation τ should reduce the experienced deformation in 3-dimensions $f_w(\tau)$:

$$A^* \in \underset{A \subseteq \mathcal{A}}{\arg\min} \, |f_w(\tau)| \tag{1}$$

The majority of literature regarding fixture design planning can be split into three main sections: firstly, design via similar part matching [6]; secondly, design through optimisation [5]; finally, design through machine learning methods [7].

The initial research in fixture design planning was centered around finding optimal plans through similarity between models. Case-Based Reasoning (CBR) is one of the earliest methods of fixture design planning and operates on the assumption that components within production Stock Keeping Units (SKUs) are similar to the extent that fixture plans can be reused [8]. CBR methods represent the oldest known autonomous method of generating new fixture designs, and has seen other variations in the form of Rule-Based Reasoning (RBR) which generates rules for finding similarities between components. CBR and RBR methods find their usefulness in fixturing tasks where the components being fixtured are variations of the same SKUs and therefore the feature extraction method can find the necessary features. The most common way of representing CBR/RBR fixture plans is through a computer language package such as that by Liqing [9]. Liqing uses an XML-based system to store the relevant fixturing plans in a database to allow for sorting and indexing in the future. Luo et al. take a different approach by transforming component shapes into graph representations where faces are nodes and vertices are edges [10]. By analysing components in the database with new components, new fixturing plans could be generated from any existing suitable plans. CBR and RBR methods, whilst initially seemed promising, rely on components having some level of similarity between them for plans to be generated, and that the initial plans stored in the database are optimal in terms of fixture placement. Additionally, McSherry notes that the effective retrieval of cases is hindered by the inseparability characteristic which can lead to misidentifying fixture plans for similar components [11].

Alongside CBR and RBR methods, optimisation-based approaches found similar appeal for fixture layout planning. Within the context of fixture layout planning, optimisation methods expand definition 1 to include constraints and pa-

rameters [12]:

$$\begin{aligned} \underset{\tau \in \mathcal{T}}{\text{minimise}} \ & (\max |f_w(\tau)|) \\ \text{s.t.} \quad & a_i \leq x_i \leq b_i \\ & c_i \leq y_i \leq d_i \end{aligned} \tag{2}$$

where:

$\mathcal{T}$ = Set of manufacturing operations performed
x_i, y_i = Coordinate positions of fixture i
a, b = Upper and lower position limits in the x-direction
c, d = Upper and lower position limits in the y-direction
$f_w(\tau)$ = Deformation in component due to task τ

Optimisation methods have a similar history to the CBR/RBR methods described previously, dating back to the 1990s [13]. With solution methods, one of the most common has been the genetic algorithm through its ability to start with initial sub-optimal solutions and explore the search space to find better solutions. Other optimisation methods that have seen interest include Particle Swarm Optimisation (PSO) [14] and active pin minimisation [2]. Xiong et al. decided to use a different approach by using the original genetic algorithm but adding the N-2-1-1 constraint that specifies the number of fixtures that can be present on a single face [12]. However, all optimisation methods can encounter sub-optimal or quasi-optimal solutions due to being stuck in local minima during the training process.

Recent methods in fixture design planning have utilised the data-based approach of machine learning techniques to find optimal fixture plans. Early work sought to mimic the methodology of CBR and RBR by using pattern recognition through neural networks to find suitable similar fixtures [15], with further work looking at using other supervised methods of machine learning [16]. However, supervised methods that rely on prior data collection fall victim to similar drawbacks as CBR and RBR due to the lack of guarantee of optimal fixturing plans. Recently, self-supervised learning (SSL) methods such as reinforcement learning (RL) have shown their ability to find fixturing plans without the need for a set of labelled data. Both Low et al. [7] and Cronrath et al. [17] use RL as their solution tool for fixture design, where Low et al. start with the maximum number of fixtures possible and let the RL agent remove a fixture at each time step and Cronrath et al. uses a quasi-optimal policy to train a digital twin using RL to find more optimal solutions. Both methods are limited in that they only allow one fixture change per train step, which can lead to long train times and inefficient policies. Furthermore, Low et al. only allow for training on individual tasks and not over a set of tasks, meaning that a new policy would need to be trained for each task. For a more comprehensive review on fixture planning, the reader is directed to [1].

B. Multi-Agent Reinforcement Learning

Multi-agent reinforcement learning (MARL) builds on the traditional RL methods by scaling the Markov decision processes (MDPs) to multiple agents. Single agent RL is characterised by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$, where at each state s_t the agent takes an action a_t which transitions the environment to the next state s_{t+1} and provides reward R_t to the agent [18]. The agent seeks to maximise the expected value of the discounted reward from the current state:

$$V(s) = \mathbb{E}_{s_{t+1} \sim P, a_t \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s \right] \tag{3}$$

where actions at each step a_t are chosen from a policy π. The agent's overall goal is to find a policy that maximises the cost found in equation 3. For multiple agents, the sequential decision making of multi-agent reinforcement learning (MARL) is now a factor of all the agents operating in the environment. This can be formulated as a *Markov* or *Stochastic game* [19], defined as a tuple $\langle N, \mathcal{S}, \{\mathcal{A}\}_{1:n}, P, \{\mathcal{R}\}_{1:n}, \gamma \rangle$. In a Markov game, each agent $n \in N$ takes an action from their action space which forms the joint action for all agents $\boldsymbol{a}_t = a_t^1 \times ... \times a_t^n, \forall n \in N$. The probabilistic state transition function P now maps the joint action and the current state into the new state $P : s_t \times \boldsymbol{a}_t \rightarrow s_{t+1}$. Similarly to the single agent problem, each agent wants to maximise their cumulative reward through the value function:

$$V^n_{\pi^n, \boldsymbol{\pi}^{-n}}(s) = \mathbb{E}_{a_{t+1} \sim P, \boldsymbol{a}_t \sim \boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t R^n_t | s_0 = s \right] \tag{4}$$

Equation 4 shows that the value function is dependent on the joint policy of all agents $\boldsymbol{\pi} = \{\pi^1, ..., \pi^n\}$. This gives way to the equilibrium condition known as Nash equilibrium.

Definition 2: For a set of agents N, an agent's policy π^n_* can be considered a best response to the set of policies for all other agents $\boldsymbol{\pi}^{-n}_* = \{\pi^1_*, ..., \pi^{i-1}_*, \pi^{i+1}_*, ..., \pi^n_*\}$ excluding agent n provided that the inequality:

$$V^n_{\pi^n_*, \boldsymbol{\pi}^{-n}_*}(s) \geq V^n_{\pi^n, \boldsymbol{\pi}^{-n}_*}(s), \forall s \in \mathcal{S} \tag{5}$$

holds true for all policies, leading to a Nash Equilibrium.

Generally, MARL literature focuses on two different types of training for the policies of the agents, shown in figure 2. Centralized training has a single policy for all agents that governs what actions each agent selects, whilst decentralized training relies on each agent having their own policy for choosing actions at each state. Sets of agents that are homogeneous can be trained in a centralized manner and then deployed decentralized, referred to as Centralized Training and Decentralized Execution (CDTE) [3]. However, if agents are heterogeneous or not in a cooperative setting, then distributed training and execution is needed as agent policies may differ in their desired action selection. This decentralized setting may be more appealing to most applications, as most settings have agents that are distributed or have a heterogeneous nature.

Due to the distribution of agents, MARL solution methods have to account for a non-stationary environment. Hu and Wellman demonstrated that Q-learning can be adapted to find a Nash equilibrium point by augmenting the Q-value with knowledge of the other agent's actions [19], a process that can be expanded with function approximators such as neural networks. Actor-critic methods have also shown promise for multi-agent systems, both with a single actor-multiple critics and multiple actors and critics [20], and MARL can also be applied to human and autonomous agents interacting through

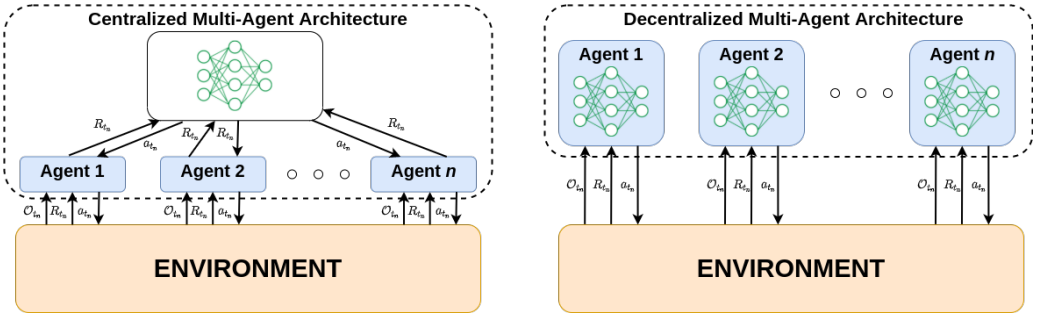


Fig. 2. Comparison between the architecture of a centralized multi-agent reinforcement learning (LEFT) and the decentralized multi-agent reinforcement learning (RIGHT) architectures

social modelling [21]. A full review of multi-agent reinforcement learning can be found in [3].

C. Research Gap

As noted in section II-A, most fixture design methods are focused on approaches that either optimise the fixture plan over a set of tasks or try to find similarities between component fixture plans. These approaches are limited in their effectiveness as group-based optimisation methods will fall into sub-optimal local minima or trade per-task optimality for a single fixturing plan. With the advent of robotic fixtures that can be reconfigured during the production process, these single-plan methods provide no benefit and can be detrimental to the component. Machine learning methods are promising, particularly the use of RL where there is not a requirement for labelled data, but applying game-theoretic MARL techniques proves difficult as rewards for fixture design would be global rather than local for each agent. In future sections, we demonstrate that our method overcomes the problem arising from optimisation methods by determining an optimal fixture plan for each task using MARL where team theory is used to enable cooperative fixture planning for multiple robotic fixtures.

III. Multi-Agent Fixture Planning

This section will explain the Multi-Robot Fixture Planner (MRFP) system using MARL. Section III-A will introduce the simplified version of RL for fixture planning, which leads into section III-B outlining the process of cooperation between agents. Section III-C will provide a complete overview of the fixture design planner in a multi-robot setting.

A. Contextual Bandits

As noted in [17], there exist a large number of systems that do not need the full MDP model for RL. These are referred to as contextual bandits, a variation of the multi-arm bandit that is present in RL literature [18]. The core idea behind the contextual bandit is that the states presented to the agent are independent of each other, meaning that the state transition probability function P no longer exists. Therefore, the tuple presented at the beginning of section II-B becomes $\langle \mathcal{S}, \mathcal{A}, \mathcal{R} \rangle$.

For fixture planning, contextual bandits represent the processes that are performed when fixtures are applied to a system, in particular drilling tasks. The set of drilling positions can be defined as the states $\mathcal{S}$, where the agent is then allowed to choose an action from its policy that represents a fixturing position. This action and state is then used in the reward function $\mathcal{R}$ that is based on the maximum deformation that the component is experiencing during this task, which the agent uses to update its policy.

In the contextual bandit setting, agents are looking to minimise the episodic regret over the total set of states:

$$\mathfrak{R}(\mathcal{S}) = \mathbb{E}\left[\sum_{s \in \mathcal{S}} R_{s,a^*} - R_{s,a}\right] \tag{6}$$

where a^* is the optimal action that yields the highest reward at each state R_{s,a^*} compared against the received reward $R_{s,a}$. However, as mentioned previously, standard RL and contextual bandit methods rely on a single action taken at each step. Whilst this approach would be suitable for fixture planning with a single agent positioning fixtures, many industrial settings have multiple fixturing elements being placed across a component, leading to an intractable solution for a single agent.

B. Decision Making for Multi-Agent Fixtures

Using multiple robotic fixtures for a single large component requires there to be cooperation between the fixtures during operation, ensuring that the agents are working together to reduce the experienced deformation. We can augment the standard contextual bandit tuple to incorporate the number of agents with their individual action spaces, giving $\langle \mathcal{S}, \mathcal{G}, \mathcal{R}, \{\mathcal{A}\}_{1:N} \rangle$ where $\mathcal{G}$ is a team of agents of size N. In this team, the agents are not homogeneous due to physical constraints of where the robots can place their fixtures, therefore their action spaces are unique to each agent. At each round, the team are presented with a drilling position and each agent $v_n \in \mathcal{G},\ \forall n \in N$ must choose a fixturing position from their action space $a_n \in \mathcal{A}_n \forall v_n \in \mathcal{G}$. The set of actions taken by the team of agents for each drilling hole is known as the joint decision rule:

$$\begin{gathered} \boldsymbol{a} = \prod_{i=1}^{N} a_i \text{ and } \boldsymbol{a}_{-n} = \boldsymbol{a} \setminus a_n \\ \forall v_n \in \mathcal{G},\ n \in N \end{gathered} \tag{7}$$

As we are now considering a team of agents acting on the component, the regret in equation 6 has to be modified:

$$\mathfrak{R}(\mathcal{S}) = \mathbb{E}\left[\sum_{s\in\mathcal{S}}\sum_{v_n\in\mathcal{G}} R_{s,\boldsymbol{a}^*} - R_{s,\boldsymbol{a}}\right] \tag{8}$$

As noted in definition 1, the goal of the fixture design plan is to minimise the expected deformation from the task being performed. This allows us to frame equation 8 as an optimisation problem that needs to be minimised for each agent policy $\phi_n \in \Phi^2$ where Φ denotes the set of policies for all agents in $\mathcal{G}$:

$$\begin{aligned} J(\mathfrak{R}) = \underset{\phi_n\in\Phi}{\text{minimise}}\ & \mathbb{E}\left[\sum_{s\in\mathcal{S}}\sum_{v_n\in\mathcal{G}} R_{s,\boldsymbol{a}^*} - R_{s,\boldsymbol{a}}\right] \\ \text{s.t.}\quad & \boldsymbol{a}^* \leftarrow \prod_{n=1}^{N} \text{argmax}_{a_n} Q(s,a_n) \\ & R_{s,\boldsymbol{a}} \sim \mathcal{R}(s,\boldsymbol{a}) \end{aligned} \tag{9}$$

To determine an equilibrium for the optimal actions that agents take the cost function can be analysed using team decision theory, a sub-discipline of game theory that deals with the control of distributed agents with a shared objective throughout the team [4].

Theorem 3.1: Let $J(\{a_n, \boldsymbol{a}_{-n}\})$ denote the cost function in equation 9 for the team decision rule $\boldsymbol{a} \in \{\mathcal{A}\}_{1:N}$. A proposed optimal team decision rule $\boldsymbol{a}^*$ can be considered a player-by-player equilibrium provided that the decision rule satisfies the inequality:

$$J(\{a_n^*, \boldsymbol{a}_{-n}^*\}) \leq J(\{a_n, \boldsymbol{a}_{-n}^*\}) \tag{10}$$

If the following criteria holds for the decision rule:

1) The reward function $\mathcal{R} : s \times \boldsymbol{a} \rightarrow R$ is convex and differentiable
2) The cost function $J(\{a_n^*, \boldsymbol{a}_{-n}^*\}$ is locally finite
3) The decision rule $\boldsymbol{a}^* \in \{\mathcal{A}\}_{1:N}$ is stationary

Then $\boldsymbol{a}^*$ can be considered the optimal decision rule.
Theorem 3.1, within the context of multi-robot fixture planning, gives that if an any agent chooses a new fixture location that is not optimal, provided that all other agents do not change their strategies, then the position would lead to an increase in the deformation in the component and thus an increase in the regret. Theorem 3.1's player-by-player equilibrium can be compared to Nash equilibrium in game theory, with the optimal decision rule satisfying Pareto optimality [4].

For the fixturing positions, there is the discussion as to whether the locations of fixel elements is continuous or discrete. Prior literature has relied on continuous spaces for optimisation methods by defining boundaries that the fixtures can be placed in [12]. However, this does not account for areas of components that cannot be grasped due to other features or perhaps material variations such as composites and metals. For this reason, a discrete action set is used for each agent that can incorporate preferences related to positions of fixturing positions and areas that cannot allow fixels to be connected.

[2]We wish to avoid confusion with the irrational number π and will use ϕ for agent policies in the rest of this paper

Definition 3: We denote the action set of agent $v_n \in \mathcal{G},\ \forall n \in \mathcal{N}$ as $\mathcal{A}_n$ as a finite discrete set of fixturing positions that each agent can take for each drilling position. The set of positions must satisfy the relationship:

$$\begin{aligned} \mathcal{A}_n := \{a \mid a \in \Gamma \text{ and } 0 < |\mathcal{A}_n| < |\Gamma|\} \\ \text{where } \bigcup_{n\in\mathcal{N}} \mathcal{A}_n \subseteq \Gamma \end{aligned} \tag{11}$$

As agents cannot place fixels in other areas, we state that the sets of actions are disjointed for all agents, such that the intersection of all agents $\bigcap_{n\in\mathcal{N}} \mathcal{A}_n$ is the empty set $\emptyset$.
For each drilling hole $s \in \mathcal{S}$, the agents select an action $a_n \in \mathcal{A}_n$ that corresponds to a fixturing position based on their policy $\phi(\cdot|s)$. The joint decision rule defined in section III-B as $\boldsymbol{a}$ denotes the position of all fixtures on a component, which is translated into a reward using the reward function $\mathcal{R} : s \times \boldsymbol{a} \rightarrow R_{s,\boldsymbol{a}}$.

Choosing a useful and meaningful reward function can be difficult for RL problems. Whilst traditional literature uses binary rewards such as $r \in [0,1]$, defining what deformation limit constitutes to a reward can vary between components. Additionally, as noted in theorem 3.1, the reward function must be convex to facilitate team-based learning and show that there is a minimum point to optimise towards. This can prove difficult for most solutions where the optimal reward $R_{s,\boldsymbol{a}^*}$ is unknown or undefined. The optimal reward can be defined as the mean of the reward function $\bar{\mathcal{R}}$ [22], over a finite set of states T, which can reduce equation 8 to:

$$\begin{aligned} \mathfrak{R}(\mathcal{S}) &= \mathbb{E}\left[\sum_{s\in\mathcal{S}}\sum_{v_n\in\mathcal{G}} \bar{\mathcal{R}} - R_{s,\boldsymbol{a}}\right] \\ &= TN\bar{\mathcal{R}} - \mathbb{E}\left[\sum_{s\in\mathcal{S}}\sum_{v_n\in\mathcal{G}} R_{s,\boldsymbol{a}}\right] \end{aligned} \tag{12}$$

This reduction allows us to redefine the cost function in equation 9 as a maximisation problem which are more common in RL as the first term in equation 12 is constant, leaving us with the objective function for training as:

$$\max_{\phi\in\Phi} \mathbb{E}_{\boldsymbol{a}\sim\Phi}\left[\sum_{s\in\mathcal{S}}\sum_{v_n\in\mathcal{G}} R_{s,\boldsymbol{a}}\right] \tag{13}$$

This objective function allows us to make some specific generalisations for the multi-agent team theory solution:

1) The problem can be interpreted as maximising the sum of rewards for all agents, known as the return of a policy
2) The inequality in theorem 3.1 still holds, provided equation 13 remains the objective during training
3) The reward function must still be differentiable, but now can be determined as concave for a maximisation problem

For a concave optimisation problem, the agents should receive greater reward as they bring the deformation closer to zero. A Gaussian reward function can be used:

$$\begin{aligned} \mathcal{R}(s,\boldsymbol{a}) = \ & \exp\left[-\frac{d_x^2}{2\sigma_x^2} - \frac{d_z^2}{2\sigma_z^2}\right] \\ & \text{s.t. } d_x, d_z \in \mathbb{R}^2_{>0} \\ & \mathcal{R} \in [0,1] \end{aligned} \tag{14}$$

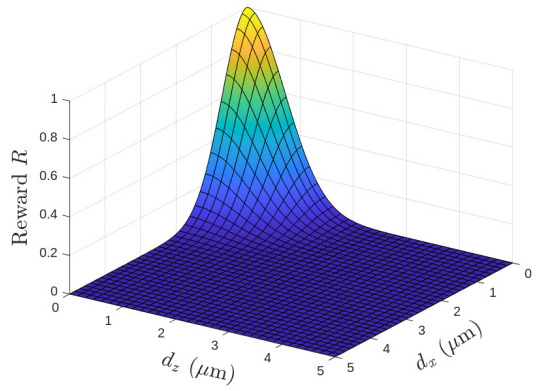


Fig. 3. The multi-variable Gaussian reward function. The values of d_x and d_z are taken from the FEA solver the computes that deformation in the x and z directions respectively. This deformation then is converted into a reward R that is returned to the agents

where:

d_x, d_z = Maximum deformations in the x, z directions
σ_x, σ_z = Variables determining the width of the curve

The z-direction remains a requirement as it is the direction in which the drilling is being performed. As for the x-direction, this corresponds to the longitudinal shear direction when the component is supported, and adding a third dimension to the reward function would over-constrain the reward function. Whilst a single dimension for deformation measurement could be used, many applications of deformation control require maintaining two dimensions within tolerance levels. The mean reward $\bar{\mathcal{R}}$ can be calculated over the function's measurement domain $U \in \mathbb{R}^2_{>0}$, shown in figure 3:

$$\bar{\mathcal{R}} = \frac{1}{\text{Area}(U)} \int_U \mathcal{R}(s, \boldsymbol{a}) dA \tag{15}$$

By definition, however, the mean of the reward function over the entire domain $\lim_{d_x, d_y \to \infty} \bar{\mathcal{R}} = 0$. To mitigate this, we can restrict the quasi-optimal reward to the domain within the tolerance for crack propagation within thin-walled composite materials [23].

Lemma 3.2: Let ζ represent the tolerance limit for crack propagation of 1.5μm. The quasi-optimal reward $\bar{\mathcal{R}}$ is therefore $\int_U \mathcal{R}(s, \boldsymbol{a})\ dA \approx 0.7331,\ U \in [0, \zeta]^2$.

C. Multi-Robot Fixture Planning

For training the multi-robot fixture planning system, each robot has a policy $\phi_n \in \Phi$ that is trained on a set of drilling positions. The algorithm of choice for this implementation is Nash-Q learning due to its ability to converge to optimal solutions in normal form games with discrete action spaces [19]. Compared to other MARL algorithms, Nash-Q remains best suited for this application as it can achieve human-level performance compared to many other learning algorithms [24]. Additionally, Nash-Q is an off-policy approach for RL problems, meaning that it can use any data that has been collected to train the model. Furthermore, this process is data efficient when compared against policy gradient methods and can learn from fewer samples compared to on-policy methods, which is beneficial for agents that are trained through fictitious play. Another benefit of using Nash-Q as opposed to other popular RL algorithms, such as policy gradient (PG) methods, is the reduction of uncertainty in action selection. PG methods generate action probabilities through a stochastic policy, leading to policies not being repeatable and increasing the uncertainty in action selection which would be a cause for concern within aerospace manufacturing. This is particularly true in Bayesian games, a subset of Markov games where the players only possess partial information about the actions of the other players [25]. As Nash-Q allows agents to build a policy that are capable of mitigating uncertainty in complex games and reducing the amount of data collection is required for training, this makes our approach most appropriate for determining optimal robotic fixture placement. For a more detailed comparison and examination of on-policy versus off-policy methods, readers are encouraged to consult [26].

We can rewrite the equation in theorem 3.1 with the maximisation approach in equation 13 to determine Nash equilibrium Q-values for each agent at each state in the game:

$$Q(s, \{a^*_n, \boldsymbol{a}^*_{-n}\}) \geq Q(s, \{a_n, \boldsymbol{a}^*_{-n}\}) \tag{16}$$

Additionally, the lack of state transitions in contextual bandits means that the performance of other algorithms is dictated by the exploration-exploitation trade-off, which is a major field of research with RL literature and beyond the scope of this work [18]. The training cycle is shown in figure 4, where the agents each observe a drilling task and contribute to the joint action which yields the reward based on the deformation. The training approach can either be done using a Q-table or by function approximation using neural networks, both of which will be shown in section IV. Algorithm 1 shows the overall training approach for the multi-robot fixture planning method using a neural network approximator. As the agents are non-homogeneous, each agent has its own multi-layer perceptron (MLP) policy network that they update in batches. During training, the agents engage in fictitious play to train their policies. For a broader overview, further details and in-depth technical requirements regarding creating and training multi-agent systems, readers should consult [20].

IV. Results

This section of the paper will outline the experimentation and results for evaluating the multi-agent reinforcement learning approach for robotic fixtures. To benchmark our approach, this section will cover two extensive case studies that are common within fixture design planning and a conceptual approach: section IV-A will cover a repeated matrix game example on a single drilling task for a wing panel, and section IV-B will expand this into optimising over a set of holes for an entire wing span. Section IV-C will introduce an ablation study into optimising for wing spar drilling to demonstrate the effectiveness of this method. For all experiments, component models and drilling positions were provided by Airbus UK and integrated into a simulation environment. Component characteristics (thickness, materials etc.) were kept constant

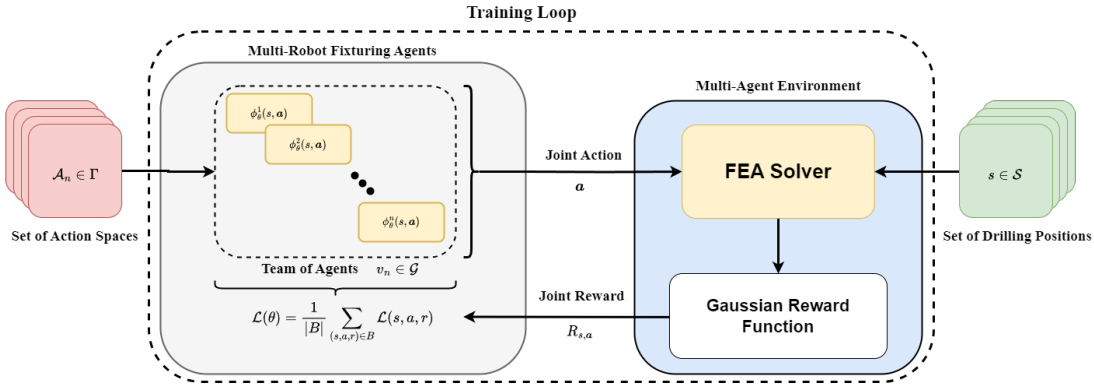


Fig. 4. An outline of the training process for the multi-robot fixture planner. Each agent policy ϕ maps an action to a drilling position based on the joint action learning process. The FEA solver uses the drilling positions and the joint action to generate a deformation profile, which the reward function uses to generate a reward for all the agents to optimise their policies

Algorithm 1 Multi-Robot Fixture Planner Training Loop

Require: $T > 1$ ▷ Require more than 1 episode
Require: $\mathcal{S} \neq \emptyset$ ▷ Set of non-empty drilling positions
Require: $v_n \in \mathcal{G}$ ▷ Set of agents
  $\phi_n \leftarrow$ Initial Bandit Policy $\forall v_n \in \mathcal{G}$
  $t \leftarrow 0$ ▷ Total number of steps taken
  $\mathcal{T}_n \leftarrow \emptyset$ ▷ Empty set of agent trajectories
  while $T < T_{\text{max}}$ **do**
    for $s \in \mathcal{S}$ **do**
      $\boldsymbol{a} \leftarrow \emptyset$ ▷ Initialise empty joint action set
      for $v_n \in \mathcal{G}$ **do**
        $a_s \leftarrow \begin{cases} \underset{a,s}{\text{argmax}}\ Q^n_{t-1}(s,a) & \Pr(1-\varepsilon) \\ \text{random action} & \Pr(\varepsilon) \end{cases}$
        $\boldsymbol{a} \leftarrow \boldsymbol{a} \cup a_n$ ▷ Create joint action set
      end for
      $R_{s,\boldsymbol{a}} \leftarrow \mathcal{R}(s,\boldsymbol{a})$ ▷ Reward from max deformation
      $\mathcal{T}_n \leftarrow \mathcal{T}_n \cup \{s, a_n, R_{s,\boldsymbol{a}}\}, \forall v_n \in \mathcal{G}$
      for $v_n \in \mathcal{G}$ **do**
        $B \leftarrow$ Random batch of trajectories from $\mathcal{T}$
        for $\langle s, a_n, R_{s,\boldsymbol{a}} \rangle \in B$ **do**
          $Q^n_t(s,a) \leftarrow \text{NashQUpdate}(Q^n_{t-1}, \langle s, a, R \rangle)$
        end for
        $\mathcal{L}(\theta) \leftarrow \frac{1}{|B|} \sum_{\langle s,a,R \rangle \in B} \mathcal{L}(Q^n_t, Q^n_{t-1})$
        Adam update $\phi_{n,\theta}$ with $\mathcal{L}(\theta)$
      end for
      $t \leftarrow t+1$ ▷ Increase the time step counter
    end for
    $T \leftarrow T+1$ ▷ Increase the episode counter
  end while

across experiments, with the FEA model being implemented within MATLAB. Our hyperparameters were chosen from academic literature [19], [20] and are detailed in table II. For our test bed, we consider two components

A. Repeated Matrix Game For Fixture Placement

To evaluate the player-by-player equilibrium for optimal actions, we first test the multi-agent reinforcement learning approach on a single drilling position for a wing panel, a representation of which is shown in figure 7. We will test

TABLE I
COMPARISON OF POLICY TYPES FOR NASH-Q LEARNING

Policy Method	Training Time (s) $n=2$	$n=3$
Q-Table	83,189 ± 876	87,810 ± 127
MLP	60,800 ± 1,129	63,220 ± 1,281

the ability of $n \in [1,2,3]$ agents to learn optimal fixturing positions, whilst examining the presence of an equilibrium position for a 2-player game. A drilling position is selected on the wing panel based on the framework outlined in [27], and a payout graph is constructed for 2 agents and a trained response for 1-3 agents is created in a similar manner to the multi-armed bandit model [18].

In figure 5a, the action numbers for each agent correspond to a fixture on the panel in figure 7, where the payout is the outcome from the Gaussian reward function. As shown in the figure, there is a definite position of equilibrium that satisfies the conditions outlined in theorem 3.1, indicating the presence of a player-by-player equilibrium for the multi-agent fixture design problem at the point where each agent chooses their best action. When the agents are trained in simulation to find the optimal positions for a single drilling hole in the method of the multi-armed bandit [18], the results of which are shown in figure 5b, we see that 2 agents outperform a single fixturing agent and reach the equilibrium action set found in figure 5a. Furthermore, the use of 3 fixturing agents compared to 2 improves the performance and reduces the deformation further.

However, whilst the Nash-Q learning method is capable of finding equilibrium solutions on single state repeated matrix games, there are two major limitations to this method. Firstly, compared to the quasi-optimal reward outlined in lemma 3.2, the agents under perform and don't reach an optimal solution. Secondly, as noted in [19], the space complexity of the Nash-Q learning algorithm is $n|\mathcal{S}| \cdot |\mathcal{A}|^n$ where n is the number of Q-tables. As the number of possible global actions is constant and the number of Q-tables is governed by the number of agents, the space complexity is exponential to the number of agents and linear with regards to the number of drilling positions. A function approximator can accommodate for this curse of

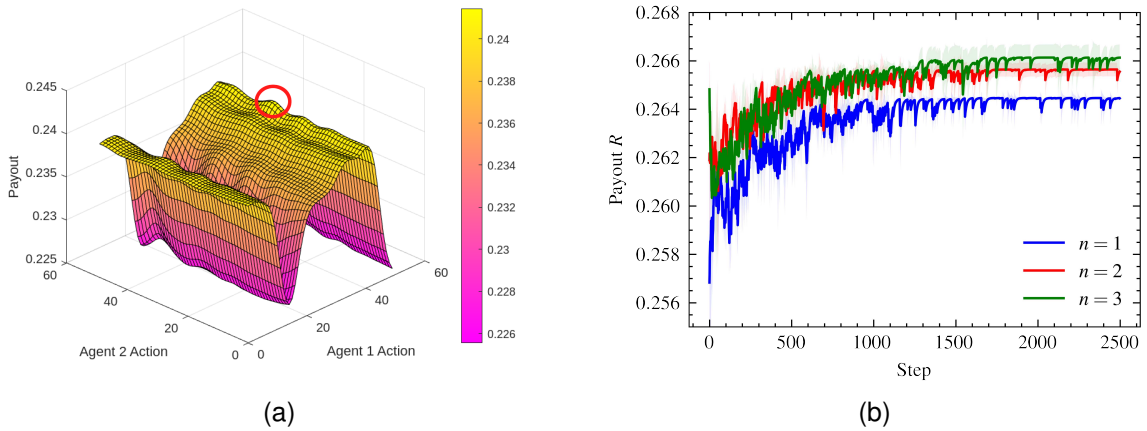


Fig. 5. Results from the n-player game scenario for the wing panel: **(a)** An equilibrium plot for the 2-player game that demonstrates the existence of a player-by-player equilibrium, marked with the red circle; **(b)** The training curves for $n \in \{1, 2, 3\}$ agents demonstrating the improving performance of increasing the number of agents and the presence of an equilibrium between them

dimensionality in fixture design by modelling the Q-table as a neural network with an input of the observed state and an output of the fixture position. This neural network mitigates the space complexity issue of the base Nash-Q algorithm by removing the complexity completely with a neural network, but also allows the use of GPU acceleration for model training and the ability to use parallel processing with multiple compute units. In table I, the Q-table approach from this section is compared against an MLP policy trained on the same drilling holes for with a multi-agent setting with $n \in \{1, 2\}$. When comparing a MLP policy function against a Q-table policy function, the training time is vastly improved due to the use of parallel computation and the MLP policy benefits from having its network stored on the GPU memory rather than in the slower system memory. These reasons necessitate the use of a function approximator for scaling the number of agents to ensure the optimal design of fixture layout plans.

B. Multi-Robot Fixture Planner For Wing Panels

The next set of experiments will be concerning the optimisation of wing panel fixture positions over a set of holes. We will compare our method against the current state of the art for wing panel fixture positioning mentioned in [12]. As in section IV-A, the wing panel model is used for experimentation. The agents are trained in PyTorch and MATLAB on a single computer with an Intel Core i9-10920X CPU and Nvidia GeForce RTX 3080 GPU, where 10 runs are trained with the average presented alongside the standard deviation from this average. Full experimentation details can be found in table II, including the hyperparameters used in creating the neural network approach. To compare directly against other methods, our wing panel was scaled to the same approximate dimensions as each of the components that are shown in table III.

The results of the training step can be found on the first row of graphs in figure 6 where some initial observations can be made. In figure 6c, we can see that only fixture sets 5 and greater are able to improve on the quasi-optimal policy based on the deformation tolerance. Additionally, the performance benefits of having 11 fixturing elements as opposed 9 elements does not offset the cost of adding the additional extra fixturing elements. Another benefit of this training method is the low variance in the final results, shown as the shaded regions in figures 6a to 6c, indicating that the training results are repeatable even in the face of uncertainty.

For evaluating the performance of the agent sets relative to state of the art methods, we can examine the maximum

TABLE II
TRAINING DETAILS FOR THE WING PANEL AND WING SPAR

Parameter	Value
Number of Episodes	100
ϵ-decay Starting Value	0.9
ϵ-decay Ending Value	0.05
ϵ-decay Rate	3500
Q-Value α Parameter	0.8
Learning rate	1×10^{-4}
Number of drilling holes	500
Batch Size	64
Reward Function Directions	x and z
Skin Panel Thickness	$5mm$
Spar Thickness	$\approx 7mm$
Material type	Isotropic Aluminium composite
Number of trials	10 per agent set

TABLE III
COMPARISON OF METHODS FOR FIXTURE PLANNING FOR PANEL-STYLE COMPONENTS AGAINST THE PROPOSED MARL APPROACH

Method	Component	Fixtures	$d_{\max}$ (μm)
N-2-1-1 Genetic Algorithm [12]	Thin-Walled Panel	3	5.68 $\pm$ 2.92
Particle Swarm Optimisation [14]	Thin-Walled Frame Panel	5	21.5 $\pm$ 4.79
Colony Optimisation Algorithms [28]	Thin-Walled Bracket	6	3.61 $\pm$ 0.09
Supervised ML Methods [16]	Thin-Walled Panel	5	126.4 $\pm$ 0.6
Active Pin Maximisation [2]	Automotive Panel	36	41.8 $\pm$ 111
MARL Fixture Planner	Aircraft Wing Panel	**5**	**1.02 $\pm$ 0.95**

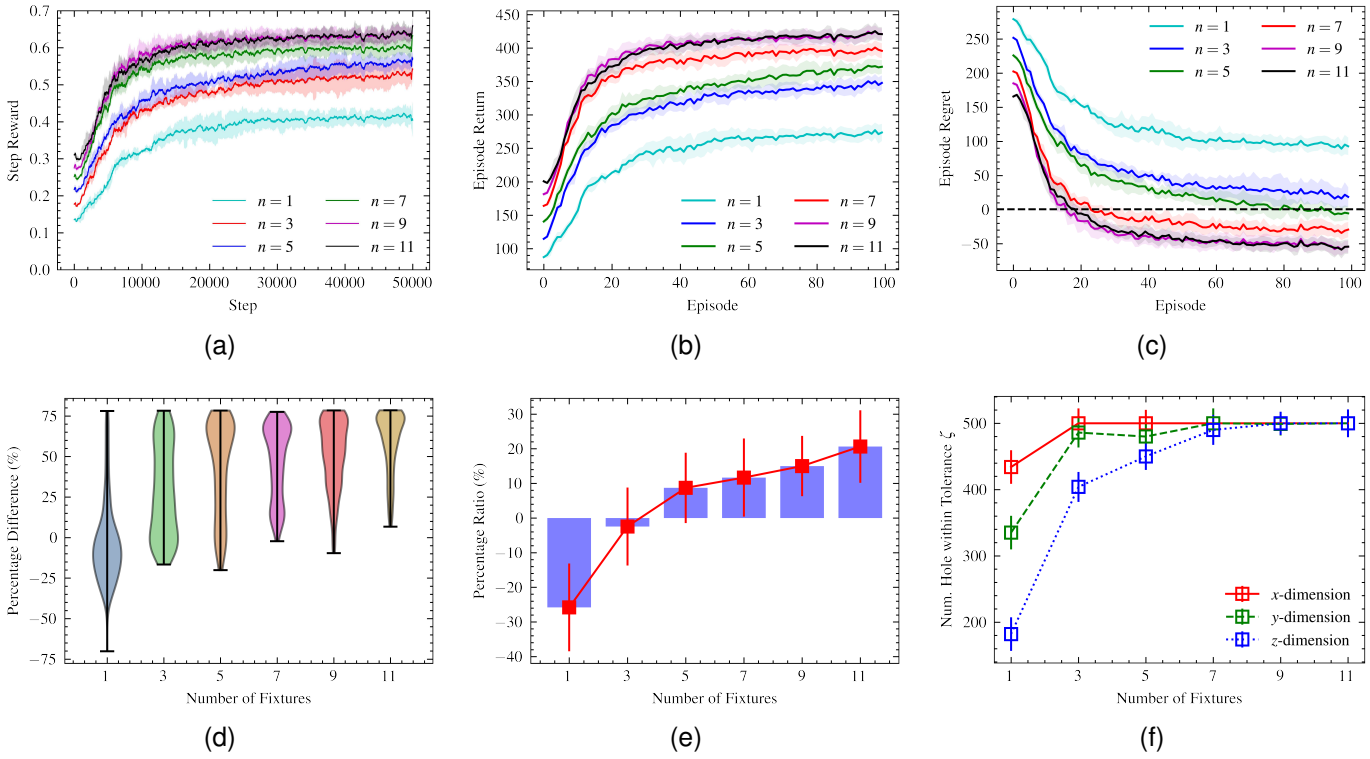


Fig. 6. Results of the training and evaluation of the wing panel multi-agent fixture planner, the top row being training results and bottom row being evaluation results: **(a)** The step reward of the agent sets; **(b)** The episodic return of the agent sets which should be viewed alongside **(c)** the episodic regret of the agent sets with the zero line indicating the relative performance to the quasi-optimal reward; **(d)** the distribution of the holes and their percentage improvement relative to the quasi-optimal reward; **(e)** The percentage improvement over the entire set of holes relative to the quasi-optimal regret and **(f)** number of holes within the tolerance limit ζ for the set of agents

deformation $d_{\max}$ experienced by the panel and compare it against recent work in the field. Due to different papers using a variety of number of fixtures, we will take the average of the papers for using our analysis and use the average deformation across all holes during evaluation. The comparisons, shown in table III, demonstrate the ability of our method to outperform other data-driven ML methods, particularly those that rely on pre-trained data-sets and supervised learning. Additionally, our method displays significant improvement over two popular optimisation methods, the genetic algorithm and particle swarm optimisation. Compared to these methods, each agent in our method has an individual policy that contains all the information it needs regarding actions and states. This would mean that, at train time, if there were any specific criteria a single agent would need - variable force control, illegal actions due to component properties - agents can be given specific encoded parameters, which can influence their individual decision making.

Another point to analyse is the performance per hole, shown in the figures 6d-6f. For examining these holes, we refer to the approximate tolerance for deformation within aerospace components, denoted as ζ in lemma 3.2. In figures 6d and 6e, we see the distribution of percentage difference from ζ per hole and the overall percentage ratio based on the regret respectively. Based on these graphs, it could be concluded that using 5 fixturing elements would be sufficient to see an improvement in the fixture plan. However, figure 6f indicates that to get all holes within tolerance for all three dimensions, 7 agents are needed to operate within the 5th-percentile. This figure also demonstrates an unintended benefit of our approach: as mentioned in table II, the reward function $\mathcal{R}$ only considered the maximum deformation in the $x-z$ directions as to minimise the shear deformation that occurs when drilling tasks are performed. Our approach was able to minimise deformation in all 3 directions and enable tolerance management across an entire set of holes for a wing panel with a single policy for each agent.

C. Wing Spar Grasping Optimisation

To further demonstrate the capability of the multi-agent fixture planning approach, another aerospace component can be tested. Continuing with the theme of wing-box assembly, this ablation study is centered around the front wing spar, a representation of which is shown in figure 9. The training approach is the same as that for the wing panel with different drilling positions for the spar drilling points, but for evaluation there are some different parameters. Firstly, there exists very little literature on wing spar deformation due to drilling as a result of the wing spar being a more complex component that is unique to aerospace assembly systems. Secondly, wing spar surface deformation tolerances are lower compared to the wing panel by reason of them being internal to the wing box and not subjected to airflow across its surface. Thirdly, wing spar crack propagation occurs when surface deformation expands when

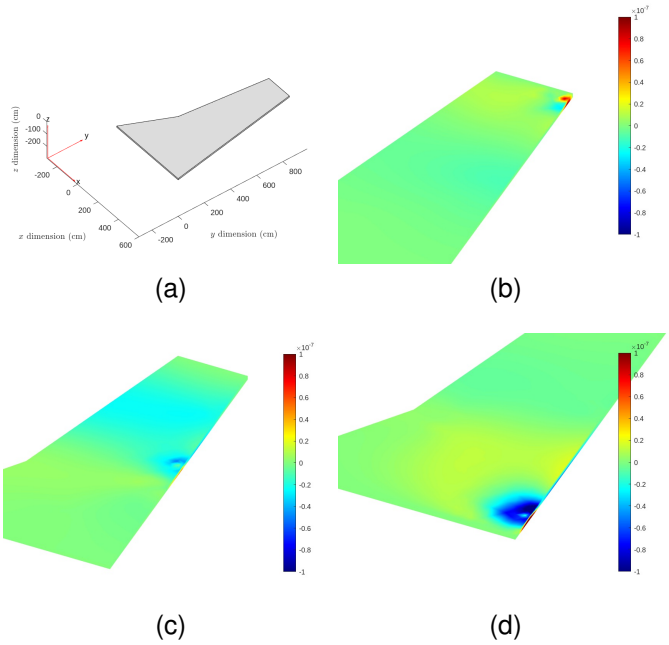


Fig. 7. **(a)** Representative model of the wing spar used in experimentation; **(b)-(d)** FEA plots of the deformation across the panel for three holes across the leading edge of the wing panel, units are in meters

under loads during flight, therefore the tolerances are placed below the maximum surface deformation limit to reduce the probability of cracks in the structure [29]. Therefore, during evaluation, we can consider the tolerance band limit ζ to be 3μm as this is a factor under the maximum tolerance of deformation cracks forming before failure in non-thin-walled structures [30].

The results of the training stage, shown in figure 8a, showcase that the multi-agent fixture planning approach is also capable of learning optimal positions for fixtures on a wing spar. This modularity showcases the strength of the multi-agent fixture planner as new agents in different locations can be added to the system without risking the prospect of them finding local optima in their search space. Another observation is that the improvement in performance when increasing the number of agents is greater for the wing spar compared to the wing panel. A reason for this is the higher complexity of the geometry of the wing spar, making it require more fixtures during the drilling process. However, this geometry is the reason for the performance shown in figure 8b. Whilst all the sets of agents are able to keep the number of holes within the 5th-percentile, not all three dimensions are maintained to within tolerance and there exists some uncertainty. A reason for this could be that, due to the geometry of the spar, other grasping locations need to be fixtured such as the bottom of the spar and on the extrusions that are present. Another possible explanation is the presence of multiple equilibria for fixture placements on the wing spar, a problem that can be mitigated with multi-objective methods [31].

V. Discussion

Over the three sets of experiments provided in this paper, we have demonstrated the capability of our MARL framework combined with team theory to out-perform traditional fixture planners. The framework provides a scalable solution with no hard requirements regarding the number of robotic fixtures that are being used and is capable to find global optimal solutions through individual robot collaboration. In all studies, each agent has a single policy that is capable of determining optimal positions for each drilling task in a manufacturing process and isn't restricted to a single sub-optimal fixturing plan for a component. This allows individual agents to manage their own policies which can enable different robot fixtures to be used to best suit the manufacturing needs. A major accomplishment of this framework is the ability to surpass the colony optimisation method put forward in Ramachandran et al. [28], achieving a 3 times improvement in deformation reduction. What's more, our framework provides optimal performance across different aerospace components, evidenced through our experiments on two different components in the wing box assembly process.

Another benefit of our approach is the scalability of the proposed framework. As noted in section IV-A, the complexity of Nash-Q learning is $\mathcal{O}(n|\mathcal{S}| \cdot |\mathcal{A}|^n)$ due to the number of Q-tables needed to be maintained. When compared directly against the methods referenced in table III, the exponential complexity of vanilla Nash-Q with no function approximation is a worse offering than the other methods. However, when using the neural network model, as each agent only has a single MLP who's input is the location of the drilling hole and output is the fixture position, this can be scaled to any number of holes and positions at a much lower linear complexity, which can be further reduced as neural networks can be trained in parallel on GPU hardware. As mentioned in section III-C, many RL methods that are on-policy require a large number of collected data points to learn an optimal policy. This is echoed in [17], which noted that their implementation of proximal policy optimisation (PPO) failed to converge to an optimal fixture plan in $10k$ environment steps. In contrast, as shown in figure 6a and figure 6c, our MARL approach with Nash-Q was able to converge to optimal fixturing plans within approximately $7.5k$ which shows an increased data efficiency when training an ML model.

Despite the successes of our work, there are some limitations that need to be addressed. The first limitation, one which is prevalent across all fixture planning literature, is the verification of the plans outside of simulation on hardware [2], [12], [14], [16], [28]. The main reason for this limitation is two-fold: firstly, acquiring the components mentioned in both this work and the work cited prior and developing an experimental test-bed remains costly in an academic setting; secondly, measuring the deformations exhibited by all the fixture planners requires high resolution sensors. This limitation can be mitigated by using a detailed FEA solver that can provide the resolution and accuracy needed to get a realistic representation of the component during the simulation step. Furthermore, when deploying robotic hardware considerations need to be made regarding the safety of the robot decision making movement. For the decision making safety, the low uncertainty of the graphs in figure 6 indicates the agent's confidence in its actions, indicating robustness in the agent's decision making. For robot movement safety, the fixture layout

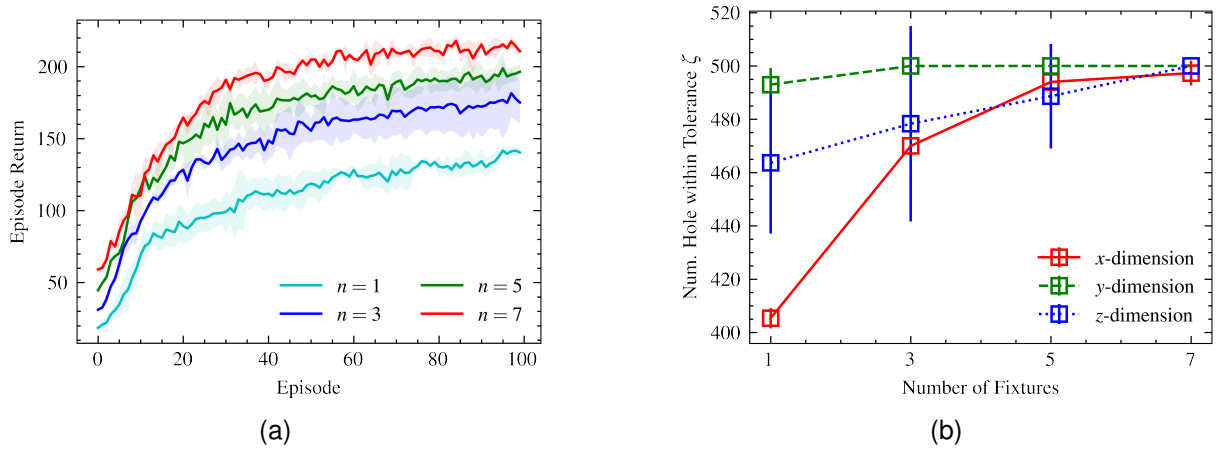


Fig. 8. Results of the training and evaluation of the front spar multi-agent fixture planner: **(a)** The episodic return at training time; **(b)** number of holes within the tolerance limit ζ for the set of agents at evaluation time

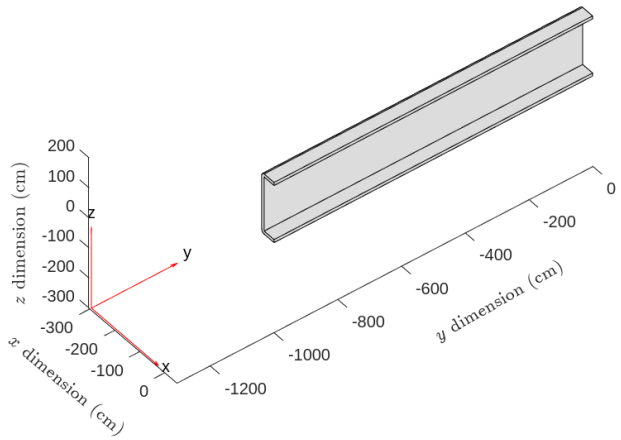


Fig. 9. Representative model of the wing spar model used in the experimentation. The actual model, supplied by Airbus UK, contains additional bends, faces and extrusion points, making it a far more complex model compared to the wing panel

planner could be combined with a path planning algorithm for multi-robot systems such as that in [32] to ensure that the robotic fixtures are moving safely without colliding with each other or human operators.

To this end, the next limitation is the choice for FEA simulator in MATLAB. Whilst we are able to account for dynamic loading and internal stresses in the component during drilling tasks, a simulator with more fidelity and the capabilities such as examining the thermo-mechanical properties and elastic bending would improve the accuracy of the FEA results. However, with a more complex FEA simulator comes higher computational complexity and hardware requirements, leading to slower training times for algorithms. This trade-off between computational cost and fidelity could be mitigated with the use of a trained surrogate model, which would improve training speed and can be customised to the desired process that is being modelled [33].

Another possible limitation of the current approach is that the agents are only evaluated on the trained holes, which for rapidly reconfigurable manufacturing may be a requirement. A mitigation for this would be to develop a multi-task framework allowing agents to learn optimal actions across out-of-distribution tasks, where learning a distribution over a set of tasks allows agents to adapt to new drilling positions.

As we mentioned in our Note To Practitioners, we have open-sourced our implementation of the MARL fixture planner on GitHub. This implementation utilises our FEA simulator in MATLAB and the agent policies in PyTorch, with a translation layer between them to allow for efficient training and reproducible results. Whilst our implementation supports the wing spar and panel that were introduced in this work, future implementations could consider expanding the library of parts to other manufacturing modalities to accelerate adoption by industrial practitioners. Additionally, whilst our agent policies are written in PyTorch and support parallelisation, the MATLAB FEA is limited to single-threaded CPU performance. For manufacturing operations, it would be beneficial to use an FEA simulator that allows for scalable training across parallel computing paradigms. Another challenge for future practical implementations is choosing the locations on the component that the fixture agents can constrain to. Current RL implementations don't allow for complex action spaces for agents, meaning that manufacturing software engineers may want to develop new software tools and libraries that build upon our framework and allow for a greater abstraction that suits their requirements.

VI. Conclusion

In this paper, we introduced a multi-agent reinforcement learning method for finding optimal fixture plans for components during drilling tasks. We outlined how robotic fixtures in a multi-agent system can constrain components and reorganise themselves to positions that are optimal for the desired task. We outlined the combination of multi-agent reinforcement learning and team decision theory into a framework that enables robotic fixtures to reconfigure themselves into optimal positions. Our method was compared directly against the state of the art methods for thin-walled panel-like components, along with a further ablation study on a more complex

geometry for an aircraft wing spar. Future work in this field would benefit from considering meta-learning as a tool for determining optimal plans on unseen tasks, the development of hardware tools that would allow the deployment and sensor-based verification of the algorithms on components and further simulation of a wider variety of components.

VII. References

[1] I. Boyle *et al.*, "A review and analysis of current computer-aided fixture design approaches," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 1, pp. 1–12, Feb. 2011, Publisher: Pergamon.

[2] J. W. Park *et al.*, "Assembly Part Positioning on Transformable Pin Array Fixture by Active Pin Maximization and Joining Point Alignment," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 2, pp. 1047–1057, Apr. 2022.

[3] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: A survey," en, *Artificial Intelligence Review*, vol. 55, no. 2, pp. 895–943, Feb. 2022.

[4] J. H. van Schuppen and T. Villa, Eds., *Coordination Control of Distributed Systems* (Lecture Notes in Control and Information Sciences 642), English, 1st ed. Switzerland: Springer, 2015, vol. 456.

[5] J. Dou *et al.*, "Machining fixture layout optimization using particle swarm optimization algorithm," in *Fourth International Seminar on Modern Cutting and Measurement Engineering*, ISSN: 0277786X, vol. 7997, SPIE, May 2010, 79970S1–79970S6.

[6] F. P. Zhang *et al.*, "Knowledge component-based intelligent method for fixture design," *International Journal of Advanced Manufacturing Technology*, vol. 94, no. 9-12, pp. 4139–4157, Sep. 2018, Publisher: Springer ISBN: 94:41394157.

[7] D. W. W. Low *et al.*, "A study on automatic fixture design using reinforcement learning," *International Journal of Advanced Manufacturing Technology*, vol. 107, no. 5-6, pp. 2303–2311, Mar. 2020, Publisher: Springer.

[8] S. H. Sun and J. L. Chen, "A modular fixture design system based on case-based reasoning," *The International Journal of Advanced Manufacturing Technology*, vol. 10, no. 6, pp. 389–395, Nov. 1995, Publisher: Springer.

[9] F. Liqing and A. Senthil Kumar, "XML-based Representation in a CBR System for Fixture Design," *Computer-Aided Design and Applications*, vol. 2, no. 1-4, pp. 339–348, 2005.

[10] C. Luo *et al.*, "A Fixture Design Retrieving Method Based on Constrained Maximum Common Subgraph," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 692–704, Apr. 2018, Publisher: Institute of Electrical and Electronics Engineers Inc.

[11] D. McSherry, "The inseparability problem in interactive case-based reasoning," *Knowledge-Based Systems*, vol. 15, no. 5-6, pp. 293–300, 2002.

[12] L. Xiong *et al.*, "Fixture layout optimization for flexible aerospace parts based on self-reconfigurable swarm intelligent fixture system," *International Journal of Advanced Manufacturing Technology*, vol. 66, no. 9-12, pp. 1305–1313, Aug. 2013, Publisher: Springer.

[13] V. Subramaniam *et al.*, "Conceptual Design of Fixtures using Genetic Algorithms," en, *The International Journal of Advanced Manufacturing Technology*, vol. 15, no. 2, pp. 79–84, Feb. 1999.

[14] S. Wang *et al.*, "Simultaneous optimization of fixture and cutting parameters of thin-walled workpieces based on particle swarm optimization algorithm," *Simulation*, vol. 94, no. 1, pp. 67–76, Jun. 2018, Publisher: SAGE PublicationsSage UK: London, England.

[15] Z. C. Lin and J. C. Huang, "The application of neural networks in fixture planning by pattern classification," *Journal of Intelligent Manufacturing*, vol. 8, no. 4, pp. 307–322, 1997, Publisher: Springer Netherlands.

[16] Q. Feng *et al.*, "Optimization of a clamping concept based on machine learning," *Production Engineering 2021*, vol. 1, pp. 1–14, Aug. 2021, Publisher: Springer.

[17] C. Cronrath *et al.*, "Enhancing digital twins through reinforcement learning," in *IEEE International Conference on Automation Science and Engineering*, vol. 2019-Augus, IEEE Computer Society, Aug. 2019, pp. 293–298.

[18] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction* (Adaptive computation and machine learning series), en, Second edition. Cambridge, Massachusetts: The MIT Press, 2018.

[19] J. Hu and M. P. Wellman, "Nash Q-Learning for General-Sum Stochastic Games," *Journal of Machine Learning Research*, vol. 4, no. Nov, pp. 1039–1069, 2003.

[20] C. Yu *et al.*, "The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games," in *36th Conference on Neural Information Processing Systems 2022*, arXiv: 2103.01955, New Orleans, USA: Neural information processing systems foundation, Mar. 2022, p. 29.

[21] E. M. de Cote *et al.*, "Learning to cooperate in multi-agent social dilemmas," in *Proc. 5th international joint conference on Autonomous agents and multiagent systems*, ser. AAMAS '06, New York, NY, USA: Association for Computing Machinery, May 2006, pp. 783–785.

[22] S. Lu *et al.*, "Stochastic Bandits with Graph Feedback in Non-Stationary Environments," *35th AAAI Conference on Artificial Intelligence, AAAI 2021*, vol. 10A, pp. 8758–8766, 2021, ISBN: 9781713835974.

[23] R. Bogenfeld *et al.*, "An experimental damage tolerance investigation of CFRP composites on a substructural level," en, *Composites Part C: Open Access*, vol. 8, p. 100 267, Jul. 2022.

[24] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015, Publisher: Nature Publishing Group.

[25] S. Zamir, "Bayesian Games: Games with Incomplete Information," in *Encyclopedia of Complexity and Systems Science*, R. A. Meyers, Ed., New York, NY: Springer New York, 2009, pp. 426–441.

[26] G. Jain *et al.*, "Recent Developments of Game Theory and Reinforcement Learning Approaches: A Systematic Review," *IEEE Access*, vol. 12, pp. 9999–10 011, 2024, Conference Name: IEEE Access.

[27] W. Hintze *et al.*, "Holistic process monitoring with machine learning classification methods using internal machine sensors for semi-automatic drilling," en, *Procedia CIRP*, Leading manufacturing systems transformation – Proceedings of the 55th CIRP Conference on Manufacturing Systems 2022, vol. 107, pp. 972–977, Jan. 2022.

[28] T. Ramachandran *et al.*, "Engine-bracket drilling fixture layout optimization for minimizing the workpiece deformation," *Engineering Computations (Swansea, Wales)*, vol. 38, no. 5, pp. 1978–2002, Aug. 2020.

[29] M. Werke *et al.*, "Machining Distortion Analysis of Aerospace Components using the Contour Method," en, Oct. 2019, pp. 216–222.

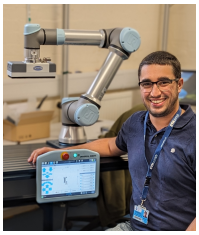
[30] F. Guo *et al.*, "Analysis on quantifiable and controllable assembly technology for aeronautical thin-walled structures," *Robotics and Computer-Integrated Manufacturing*, vol. 80, p. 102 473, Apr. 2023.

[31] S. M. LaValle, *Planning Algorithms*. Cambridge: Cambridge University Press, 2006.

[32] S. Veeramani *et al.*, "Artificial intelligence planners for multi-head path planning of SwarmItFIX agents," *Journal of Intelligent Manufacturing*, vol. 31, no. 4, pp. 815–832, 2020, Publisher: Springer US ISBN: 0123456789.

[33] J. Kudela and R. Matousek, "Recent advances and applications of surrogate models for finite element method computations: A review," en, *Soft Computing*, vol. 26, no. 24, pp. 13 709–13 733, Dec. 2022.

VIII. Biographies



Ethan Canzini obtained an MEng degree in Aerospace Engineering from the University of Sheffield in 2021 with a double minor in control and robotics. He has industrial experience in the aerospace and semiconductor sectors, specialising in autonomous systems. He is currently undertaking a PhD in Automatic Control from the University of Sheffield and is a Research Scientist at Airbus Robotics.

Dr Marc Auledas-Noguera received a BEng and MEng in industrial engineering from the Polytechnic University of Catalonia. He was awarded an MSc in Robotics in 2019 and a PhD in Robotics in 2023 from The University of Sheffield. At the time of submission he is a research associate in Robotics with the Department of Automatic Control and Systems Engineering at The University of Sheffield.

Dr Simon Pope is a Lecturer in Active Control at the University of Sheffield. He graduated with a MEng degree in Mechanical Systems Engineering in 2004 and a PhD in 2009, both from the University of Sheffield. He specialises in the application of control and systems engineering principles to a range of mechanical systems, including acoustic, vibration, material and manufacturing systems.

Prof. Ashutosh Tiwari is Deputy Vice-President for Innovation at the University of Sheffield and holds the prestigious RAEng/Airbus Research Chair. He is internationally renowned for research in digital manufacturing and works in partnership with industry to develop new techniques and solutions for digitalisation in manufacturing operations.

He has a strong track record of leading research and innovation projects across technology readiness levels, and serves on the EPSRC Strategic Advisory Team for Manufacturing and the Circular Economy.