2023 Ninth Indian Control Conference (ICC)
18 - 20 December, 2023
GITAM University, Visakhapatnam, India

A Non-iterative Spatio-temporal Multi-task Assignments based Collision-free Trajectories for Music Playing Robots

Shridhar Velhal[1], Krishna Kishore VS[2] and Suresh Sundaram[3]

***Abstract*— This paper addresses the music-playing robot problem, which is a benchmark problem for the spatio-temporal multi-task assignment problem. In the music-playing robot problem, an algorithm needs to compute the trajectories for a dynamically-sized team of robots who will play musical notes by traveling through the specific locations associated with musical notes at their respective specific times. A two-step dynamic resource allocation based on a spatio-temporal multi-task assignment problem (DREAM) has been implemented to assign robots to play a musical tune. The algorithm computes the number of robots required to play the music in the first step. In the second step, optimal assignments are computed for the updated team of robots, which minimizes the total distance traveled by the team. Even for individual feasible trajectories, the multi-robot execution may fail if robots encounter a collision. As some time will be utilized for this conflict resolution, robots may not be able to reach the desired location on time. This paper analyses and proves that if robots are operating in a convex region, the solution of the DREAM approach provides collision-free trajectories. The working of the DREAM approach is illustrated using high-fidelity simulations in a Gazebo operated using ROS2. The result clearly shows that the DREAM approach computes the required number of robots and assigns multiple tasks to robots in at most two steps. A simulation of the robots playing the 'happy birthday' is available at `https://youtu.be/XToicNm-CO8`.**

I. Introduction

Multi-robot systems have been used to execute tasks where a team of robots needs to visit a set of spatially distributed locations. The robots have been used for different purposes such as logistics [1], search [2], surveillance [3], [4], [5], and various other applications [6], [7], [8]. Robots are assigned to tasks such that they minimize some specific criteria (e.g., minimization of total distance traveled [9], completion time [10], [11], energy consumption [12]). With recent IoT and communication infrastructure developments, customers demand spatial visits with time constraints. In logistics operations, the delivery person not only needs to visit the locations but needs to visit those locations with some time constraints. The multi-task assignment problems with different temporal constraints are reviewed in [13].

The music-playing problem is the benchmark problem for spatio-temporal tasks. For executing a task, a robot needs to visit the desired location at the desired time to execute the spatio-temporal tasks. In operational research, spatio-temporal tasks are also referred as just-in-time (JIT) tasks. The promised-deliver-time (PDT) positively impacts the brand image [14]. The time-constrained problem has been extensively studied for industry automation [15], time-constrained delivery problems [16], [17], [18], and time-constrained cross-dock management [19], [20], hub-arrival-departure [21]. All the aforementioned works highlight the need and importance of spatio-temporal or just-in-time task executions. But they did not comment on the feasibility of the posed problem. For a given problem there, is any solution exists or not? The feasibility of the problem depends on the tasks and the number of agents available to execute those tasks. All the aforementioned works assume that sufficient agents are available to execute the given tasks and then solve the assignment problem. This paper highlights the theoretical features and demonstrates the application DREAM [22] algorithm for the music-playing problem to provide a non-iterative solution to guarantee feasibility.

The problem of music-playing robots is formulated by Smriti Chopra and Magnus Egerstedt in [23], [24], [25]. For playing a musical note, a robot must visit the specific location which plays that sound note. Each musical note is connected to a different spatial location. Robots must visit the prescribed note locations at specific times to play musical tunes. In this way, the problem of music-playing robots is converted to multiple spatio-temporal tasks for the team of robots. A team of robots needs to compute the assignments of robots to multiple spatio-temporal tasks, following which they can play the music. The task allocation and scheduling schemes [26], [27] have been used to schedule the tasks for multiple robots. But in these approaches tasks are handled

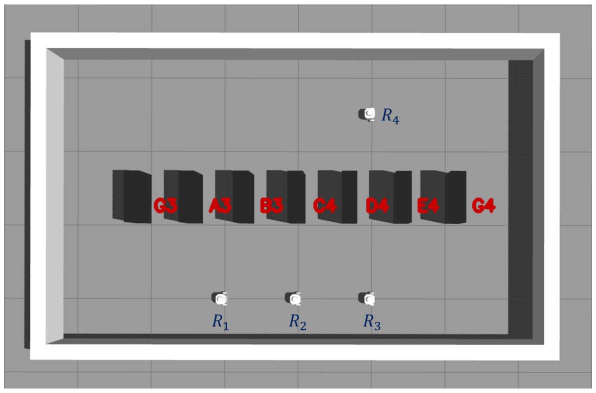

Fig. 1. Piano Setup in Gazebo for playing music using turtlebots

[1]Shridhar Velhal is a Ph.D. student in Department of Aerospace Engineering, Indian institute of Science, Bangalore, India. velhalb@iisc.ac.in
[2]Krishna Kishore VS is B.Tech student in Department of Electronics and Communication Engineering, National Institute of Technology, Tiruchirappalli, India. krishnakishorevs@gmail.com
[3] Suresh Sundaram is an Associate Professor in Department of Aerospace Engineering, Indian institute of Science, Bangalore, India. vssuresh@iisc.ac.in

2023 Ninth Indian Control Conference (ICC) | 979-8-3503-9525-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICC61519.2023.10442571

with soft temporal constraints and all tasks will not be executed at the desired time. Hence can not be applied for the exact time constraints. Also due to the exact-time constraints, time-windowed task assignment approaches can not be used. The just-in-time tasks assignment [28], [29] approaches have been employed in automobile and automation industries, but it assumes tasks are predefined and the solution requires the off-line computations. The spatio-temporal multi-task assignment (STMTA) problem has been studied in [23], [24], [25], [30], [22], [31].

For the given spatio-temporal tasks, depending on the tasks and velocity of robots, some minimum number of robots are required to execute them. If the formulated STMTA is ill-posed, the obtained solution is invalid and demands unrealistic velocities. The velocity-constrained STMTA problem is solved in [24]. A Hungarian problem [32] is solved multiple times, by increasing the robots one by one until a feasible solution is achieved (iterative method).

Recently, a non-iterative solution has been proposed using Dynamic resource allocation with a decentralized multi-task assignment (DREAM) approach [22]. DREAM solves the STMTA (Hungarian) problem, once to compute the required minimum number of robots. Then, with an updated team of robots, STMTA is solved to get the optimal solution. So with only two steps, DREAM computes the required number of robots and their optimal assignments.

In this paper, the music-playing problem is solved and the execution guarantee is proved. Once the spatio-temporal tasks are generated for a given music, the current team of robots uses the DREAM algorithm. DREAM algorithm computes the minimum number of robots required, updates the team, and with an updated team it computes the optimal assignments. Based on the assignments, the trajectory of each robot is computed. Even if all individual trajectories are feasible, one can not guarantee execution due to possible collisions among multiple robots. Also, note that reactive collision avoidance methods require some extra (and uncertain) time to resolve the conflicts. This delay may violate the temporal feasibility constraint. Hence, there is a need to guarantee that the solution has collision-free trajectories.

The main contribution of this paper is the theoretical analysis of the collision-free nature of obtained trajectories. In this paper, we prove that the obtained trajectories are collision-free for homogeneous robots operating in convex space. By following the computed trajectories, robots will not collide with other teammates and unassigned robots (which were removed from the team). The same has been proved for robots operating in the arena (simulated environment) used in the paper. The working of the music-playing robots is demonstrated using turtlebots in Gazebo simulations.

The rest of the paper is organized as follows: Section II provides the mathematical problem formulation for piano-playing robots. Section III explains the DREAM approach to solve spatio-temporal tasks. Section IV proves the collision-free nature of obtained trajectories. Section V explain simulation architecture and presents the performance of robots for playing music. Finally, Section VI concludes the paper.

II. Problem Formulation

Consider a two-dimensional arena with walls and musical strings as shown in fig 1. Consider a string placed at the center of lanes between the walls, which will generate sounds with a specific frequency whenever the robots cross that midsection of the lane. Robots will travel through the lanes and plunk the strings to play the musical notes. With this piano set-up, a musical tune is interpreted as the sequence of the musical notes to be played at specific time instants. Robots need to reach the respective note locations at specific times. This defines the spatio-temporal tasks for robots. Robots can play the music by routing through such timed positions. Next, tasks and operations for piano music-playing robots are mathematically formulated. Before that, we list the notations used in the paper.

The following symbols are used in paper: R: robot; t: time; $\boldsymbol{p}$: position; μ: sequence of tasks assigned to robot; $(\cdot)_i^R$: for ith robot; $(\cdot)_j^T$: for jth task; t_j: desired execution time of task T_j; $T_j(\boldsymbol{p}_j^T, t_j) = T_j$: j^{th} spatio-temporal task; C_{ij}^f : cost of R_i executing the T_j as a first task; C_{kj}^s: cost of robot will execute the task T_j just after the task T_k (subsequent task); δ_{ij}^f: decision variable whether robot R_i execute the task T_j as first task or not; δ_{kj}^s: decision variable whether robot execute the task T_j just after task T_k or not.

Fig 1 shows the piano arena created in the Gazebo 11, and Turtlebot3 are used as robots. The arena consists of 7 equally separated walls. Each lane is associated with a predefined specific music tune. A robot must cross the lane's midsection to play that specific note. A robot must wait at the entry point until the desired time and cross the lane at the desired time. The musical note will be played once the robot passes through that lane's midsection. A robot can cross the patch either by traveling top-to-bottom or bottom-to-top.

A. Mathematical Problem Formulation

Consider a set of N robots denoted as $\mathcal{R}$, $\mathcal{R} = \{R_1, R_2, \cdots, R_N\}$. The positions of robots R_i is denoted as $\boldsymbol{p}_i^R = (x_i^R, y_i^R)$. Robots need to play a bunch of piano notes in properly timed sequences.

1) Spatio-temporal tasks : Consider a musical note j with the position $\boldsymbol{p}_j^T = (x_j^T, y_j^T)$ and desired time t_j. To play this note, the robot has to visit a waiting area near the desired lane, wait until the desired execution time, and then pass through the desired location $\boldsymbol{p}_j^T$ at desired time t_j to play that musical note. The steps in playing notes are (a) reach to the waiting location near $\boldsymbol{p}_j^T$ (b) wait till desired time t_j (c) cross the lane

In the rest of the paper, the musical note playing is defined as a spatio-temporal task $T_j(\boldsymbol{p}_j^T, t_j)$ is referred as T_j. This paper assumes that all robots are homogeneous (same maximum velocity (v^{max}) and can play all musical notes).

To play the musical tune, a team of robots needs to play a sequence of musical notes. Let us consider that a musical tune has M notes (in general $M > N$), and a team of robots needs to execute the given M spatio-temporal task. This defines the spatio-temporal multi-task assignment problem.

The objective of the problem is to compute the assignments for the team of robots such that they will play the musical notes at respective times. One should note that for playing the musical notes, i.e., for executing the spatio-temporal tasks, some minimum number of robots is required. If the given number of robots is less than this required number, then the problem is ill-posed, and all tasks can not be executed. Hence the objective of the problem is twofold; the first is to compute the minimum number of robots required to play the given musical notes, and the second is to compute the assignments for an updated team of robots to play the given musical notes.

III. Spatio-Temporal Multi-Task Assignments

The team of robots needs to play the musical notes by traveling through the lane of respective notes at desired times. These task demands the robots to reach specific spatial locations at specific times; hence tasks are termed spatio-temporal tasks. For executing the given spatio-temporal tasks, a minimum number of robots is required. Otherwise, the problem is ill-posed, and the given team of robots cannot execute all the spatio-temporal tasks with any assignments. This minimum number of robots required depends on the velocity limit and the given spatio-temporal tasks.

We use the DREAM approach proposed in [22] to compute feasible assignments. The DREAM approach uses a two-step method in which, at first, the infeasible tasks are assigned a cost equal to a large value, and the task assignment problem is solved with a given number of robots. Now from the solution, the number of infeasible assignments is identified, and those many robots are added to the team of robots. The updated team size is the minimum number of robots, and the proof of this optimality is provided in [22]. At first, the cost function is defined for the tasks.

A. Cost Function

The cost of a spatio-temporal task is the distance that needs to be traveled by a robot to reach the task location on or before the time of the task from its previous location. For a robot executing its first task from its initial position, the cost of the first task (C^f) is the distance traveled by the robot from its current position to the task position on or before the desired task time.

First we denote $d(\boldsymbol{p}_i^R, \boldsymbol{p}_j^T)$ as the distance computed along the feasible path from point $\boldsymbol{p}_i^R$ to $\boldsymbol{p}_j^T$. This distance is computed using the A^* motion planning algorithm on the map of the arena. Now cost for the first task is defined as

$$C_{ij}^f = \begin{cases} d(\boldsymbol{p}_i^R, \boldsymbol{p}_j^T) & \text{if } \dfrac{d(\boldsymbol{p}_i^R, \boldsymbol{p}_j^T)}{V_{max}^R} \le t_j \\ \kappa & \text{otherwise} \end{cases} \tag{1}$$
$$\text{for } i \in \mathcal{I} = \{1, 2, \cdots, N\}, \quad j \in \mathcal{J} = \{1, 2, \cdots, M\}$$

where, κ is a large value.

The cost for executing the subsequent tasks (C^s) by the robot is the distance traveled by the robot from its previous task location to reach the current task location on or before the desired execution time of that subsequent task.

$$C_{kj}^s = \begin{cases} d(\boldsymbol{p}_k^T, \boldsymbol{p}_j^T) & \text{if } t_{k,j}^{min} \le (t_j - t_k) \\ \kappa & \text{if } t_{k,j}^{min} > (t_j - t_k) > 0 \\ \infty & \text{if } t_j - t_k \le 0 \end{cases} \tag{2}$$
$$\text{for } k \in \mathcal{K} = \{1, 2, \cdots, M-1\} \quad j \in \mathcal{J}$$

where $t_{k,j}^{min}$ is the minimum time required by robot to travel from the location of task T_k to task T_j and it is computed as $t_{k,j}^{min} = d(\boldsymbol{p}_k^T, \boldsymbol{p}_j^T)/V_{max}^R$

B. Optimization Problem

A task allocation algorithm assigns robots to multiple spatio-temporal tasks. A robot will execute the tasks in a sequence, and we denote the sequence assigned to robot R_i by μ_i. Here, the problem of computing sequence μ_i has been converted to compute each move of one robot from one location to another; combining all moves, one can get the sequence of tasks. Each robot computes its sequence to execute all spatio-temporal tasks while minimizing the distance traveled. The first decision variable δ_{ij}^f is used to denote that either a robot moves from position p_i^R to execute the task T_j as a first task or not. The subsequent decision variable δ_{kj}^s is used to denote that either a robot will do task T_j just after task T_k or not. The integer programming problem is defined as,

$$\min_{\delta_{ij}^f \ \delta_{kj}^s} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} C_{ij}^f \delta_{ij}^f + \sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{J}} C_{kj}^s \delta_{kj}^s \tag{3}$$
$$\text{s. t. } \delta_{ij}^f \in \{0,1\} \qquad \forall (i,j) \in \mathcal{I}\times\mathcal{J} \tag{3a}$$
$$\delta_{kj}^s \in \{0,1\} \qquad \forall (k,j) \in \mathcal{K}\times\mathcal{J} \tag{3b}$$
$$\sum_{i\in\mathcal{I}} \delta_{ij}^f + \sum_{k\in\mathcal{K}} \delta_{kj}^s = 1 \quad \forall j \in \mathcal{J} \tag{3c}$$
$$\sum_{j\in\mathcal{J}} \delta_{ij}^f \le 1 \quad \forall i \in \mathcal{I} \tag{3d}$$
$$\sum_{j\in\mathcal{J}} \delta_{kj}^s \le 1 \quad \forall k \in \mathcal{K} \tag{3e}$$

All tasks must be assigned as a first or subsequent task to exactly one robot; this constraint is given by (3c). A robot can move to at most one task location just after completing the current task, which is constrained by eq. (3d) and (3e).

C. DREAM Approach

The spatio-temporal tasks require some minimum number of robots to execute the given spatio-temporal tasks. The problem formulated in Eq. (3) solves the assignment problem for a given N number of robots. As the infeasible tasks are given a cost equal to the κ, the solution of Eq (3) may contain some infeasible assignments with a cost equal to κ. The DREAM approach computes the minimum number of robots required to execute all the given tasks (for which all computed assignments will be feasible). For clarity, the DREAM algorithm [22] is provided in algorithm 1.

Once all the feasible assignments are obtained using the 2-step solution provided by the DREAM approach, robots

Algorithm 1 Dynamic resource allocation algorithm

```
1: Initialize with $N$ robots and $M$ musical notes to be played with respective timings
2: Solve the optimization problem (eq. (3))
3: $q$ = number of assignments with cost equal to $\kappa$
4: if $q > 0$ then
5:     Add new $q$ robots in team near to the spatial location of infeasible tasks
6:     $N = N + q$
7:     Solve the optimization problem (eq. (3))
8: end if
9: feasible assignments are obtained
10: Compute the sequence of tasks assigned to each robots using the assignments
```

must execute the assigned tasks in sequence. The assignment solution provides the assignments of robots from one location to another. Augmenting the next positions to the last, one can compute the sequence of tasks assigned to each robot. The detailed trajectory generation algorithm can be found in [22].

IV. Analysis for computed trajectories

Analysis in [22] proved that the solution obtained from DREAM is optimal, and all trajectories are feasible for individual robots. But in multi-robot systems, the feasibility guarantee may fail if robots encounter any collision. If all trajectories are collision-free, execution can be guaranteed.

First, we state the existing analysis for the optimal and collision-free (non-intersecting path) solutions from [33].

Theorem 1. *(**Theorem 3.1 from [33]**) The optimal assignment using the minimum sum of distance optimization results in non-intersecting paths with the exception of the special case when a pair of robots have collinear start and goal locations.*

This theorem proves that the solution linear sum assignment problem minimizes the distance traveled and gives collision-free paths. As the paths of robots are collision-free, the obtained trajectories are also collision-free. But solution to a temporally constrained problem may have colliding paths. For example, consider fig 2 with $t_D < t_A$ (then trajectory A to D becomes infeasible), and all other trajectories are feasible. Then the solution obtained is different from the unconstrained minimum path problem, and it has colliding paths. Now, one can not comment on the collision-free nature of the trajectories of the obtained solution of a temporally constrained problem, as paths may collide with each other.

Theorem 2. *For homogeneous robots operating in a convex region, the feasible solution obtained from the DREAM algorithm gives collision-free trajectories.*

Proof. We will prove the theorem by contradiction. Without the loss of generality, we assume that the computed trajectories of robot R_1 and R_2 have conflict and will collide. Let us consider R_1 is moving from $(\boldsymbol{p}_A, t_A)$ towards spatio-temporal point $(\boldsymbol{p}_B, t_B)$ with velocity v_1 and R_2 is moving

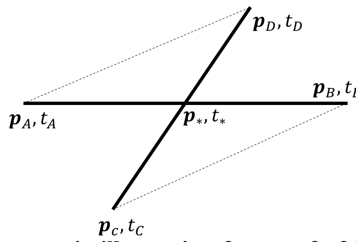

Fig. 2. Geometric illustration for proof of Theorem 2

from $(\boldsymbol{p}_C, t_C)$ towards spatio-temporal point $(\boldsymbol{p}_D, t_D)$ with velocity v_2 and while doing this the robots will collide at location $\boldsymbol{p}_*$ at time t_*. Without loss of generality, we consider $v_1 \le v_2 \le v^{max}$. This scenario is shown in Fig 2 for clarity.

As the solution is optimal, $\|\boldsymbol{p}_A\boldsymbol{p}_B\|_2 + \|\boldsymbol{p}_C\boldsymbol{p}_D\|_2$ is the minimum cost for executing tasks T_B, and T_D by robots R_1 and R_2. Now due to collision both robot reach $\boldsymbol{p}_*$ at time t_*. The remaining trajectories will be feasible for both robots from the collision points. As robots are homogeneous, if they exchange their trajectories, the trajectories still remain feasible. (Let us refer to them as alternate trajectories).

In the alternate solution, the tasks after collision points do not change the total distance that needs to be traveled by both robots. Alternate trajectories are feasible and require traveling the same distance; hence they are also optimal. i.e. $\|\boldsymbol{p}_A\boldsymbol{p}_*\|_2 + \|\boldsymbol{p}_*\boldsymbol{p}_D\|_2 + \|\boldsymbol{p}_C\boldsymbol{p}_*\|_2 + \|\boldsymbol{p}_*\boldsymbol{p}_B\|_2$ is the minimum cost for executing tasks T_B, and T_D by robots R_1 and R_2

In alternate optimal solution R_1 travels along the path $\boldsymbol{p}_A - \boldsymbol{p}_* - \boldsymbol{p}_D$, but there exists direct path $\boldsymbol{p}_A - \boldsymbol{p}_D$. By the triangle inequality, direct path from $\boldsymbol{p}_A$ to $\boldsymbol{p}_D$ is shorter. On direct path $\boldsymbol{p}_A - \boldsymbol{p}_D$, robot R_1 can reach location $\boldsymbol{p}_D$ before the desired time t_D. i.e., the direct path is feasible. Hence the cost of the alternate solution is not optimal.

This contradiction proves that the assumption of the existence of a collision point is incorrect. Thus it is proved that the optimal solution obtained from the DREAM approach gives collision-free trajectories. $\square$

Corollary 2.1. *For homogeneous robots operating in the convex region, if the robot follows the trajectory computed by the DREAM algorithm, then the robot will not collide with any unassigned robots.*

It can be proven with a case of $v_1 = 0$ (unassigned defender executes the fictitious task at the same location and travels with zero velocity) in the proof given for Theorem 2.

Corollary 2.2. *For the music-playing problem defined in this paper, the computed trajectory of robots is collision-free.*

Proof. Let us divide the arena into three regions, Ω^1 (a region above the lanes), Ω^2 (a region below the lanes), and Ω^3 (remaining region of lanes). Without loss of generality, we assume that a robot R_i is initialized in Ω^1. As the robot is in Ω^1, it will be asked to visit the entry location of the lane which lies in Ω^1. The region Ω^1 is convex; using the Theorem 2, a robot will have a collision-free trajectory.

For playing music but R_i has to wait at the entry point till the desired time and then possess through lanes, i.e., Ω^3

and goes to region Ω^2. From the region, Ω^1, R_i can enter region Ω^3 only for playing music and only at desired task time. Only one robot is assigned to play a musical note, so no two robots will enter the lane at the same time. Hence robots won't collide with any other robots in Ω^3. Once the robot reaches the other end of the lane, it enters the region Ω^2. For the next task, it will be asked to visit the entry point of the assigned lane, which lies in Ω^2. Region Ω^2 is convex; hence, the robot will have a collision-free trajectory in Ω^2.

The robot will have a collision-free trajectory whenever it is in the region Ω^1,Ω^2, and Ω^3. Hence, a robot has a collision-free trajectory for the entire arena. □

A. Complexity analysis

Consider $\mathcal{O}(\cdot)$ denotes the upper bound on the complexity of the algorithm, and $\Omega(\cdot)$ denotes the lower bound on the complexity of the algorithm, $\mathcal{H}(n)$ denotes the Hungarian problem of size n. For the Hungarian algorithm, the complexity is upper bounded by $\mathcal{O}(n^3)$ and lower bounded by $\Omega(n^2)$. For M tasks with N required robots, the complexity of DREAM is denoted by C^{DREAM}, and the complexity of the iterative algorithm ([24]) is denoted by C^{iter}.

$$\begin{aligned} C^{DREAM} &= \mathcal{H}(M) + \mathcal{H}(M+N-1) \\ &= \mathcal{O}\left((M+N-1)^3\right) \\ C^{iter} &= \mathcal{H}(M) + \mathcal{H}(M+1) + \cdots + \mathcal{H}(M+N-1) \\ &= \mathcal{O}\left(N(M+N-1)^3\right) \end{aligned}$$

The reduction in complexity is defined to compute the lower bound on reduction in computation as,

$$\begin{aligned} C^r &= C^{iter} - C^{DREAM} \\ &= \mathcal{H}(M+1) + \mathcal{H}(M+2) + \cdots + \mathcal{H}(M+N-2) \\ &\geq \Omega\left((M+1)^2\right) + \cdots + \Omega\left((M+N-2)^2\right) \\ &\geq \Omega\left((M+N-2)^3 - (M)^3\right) \end{aligned}$$

The complexity of the DREAM algorithm is cubic, and it is N is times lower than the iterative algorithm. Furthermore, the iterative algorithm requires at least C^r complexity more than DREAM.

V. Simulation and Results

The DREAM algorithm has been used to play piano music. This system is implemented and tested with the help of ROS2 in the Gazebo simulation environment. The simulations are conducted in Ubuntu 20.04 system with a 3.20GHz processor and 16 GB RAM.

A. Performance Evaluation

The Piano setup is shown in Fig. 1. The team of 4 robots plays a happy birthday tune in the piano arena to illustrate the working of the proposed approach. A total of 7 different musical notes are used. Here, the gap between the two walls is 0.5m, and the lane length is 0.4m, but for turning robots (considering the waiting area), the practical distance that needs to be traveled is 0.8 m. The turtlebots (with a max velocity of 0.5 m/sec) are used as robots to travel in the arena

Task No	Note	time (sec)
1	G3	105
2	G3	113
3	A3	119
4	G3	124
5	C4	133
6	B3	142
7	G3	157
8	G3	163

Task No	Note	time (sec)
9	A3	168
10	G3	172
11	D4	183
12	C4	192
13	G3	206
14	G3	214
15	G4	221
16	E4	229

Task No	Note	time (sec)
17	C4	235
18	B3	239
19	A3	250
20	G4	260
21	G4	270
22	C4	280
23	D4	290
24	D4	300

TABLE I: Spatio-temporal tasks for 'Happy birthday' tune

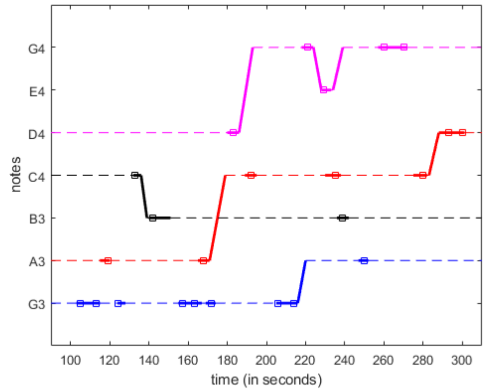

Fig. 3. Operations of robots while playing music

and play music. As the distance that needs to be traveled is large compared to the velocity of the robots, the musical tasks are given for execution by scaling the time ten times.

A total of 24 musical notes are required to play the tune, and the spatio-temporal tasks considered are given in table I. The attached video demonstrates the music-playing robots.

Once all tasks are received, the robots use the map of the piano arena to compute the distances and then compute the cost matrices. When only one robot is used to solve STMTA, it demands three extra robots. So, total of four robots has been used to play the desired tune. The computed trajectories for the robots are, for robot 1, $\mu_1 = \{T_1, T_2, T_4, T_7, T_8, T_{10}, T_{13}, T_{14}, T_{19}\}$; for robot 2, $\mu_2 = \{T_3, T_9, T_{12}, T_{17}, T_{22}, T_{23}, T_{24}\}$; for robot 3, $\mu_3 = \{T_5, T_6, T_{18}\}$, for robot 4, $\mu_4 = \{T_{11}, T_{15}, T_{16}, T_{20}, T_{21}\}$.

The video `https://youtu.be/XToicNm-CO8` shows the operations of robots to play the 'happy birthday' tune. Fig 3 shows the operations of robots. Four different colors are used for the four robots. The solid line denotes the robots in motion, and the dashed lines denote the robots waiting. One can observe that four robots are required to execute tasks from 230 sec to 250 sec. If any robot is absent, then all notes can not be played. It shows that the team size computed by the DREAM (i.e., 4 robots) is a necessary and sufficient team size to execute the given spatio-temporal tasks.

DREAM requires solving the Hungarian problem only twice for $N = \{1, 4\}$, while iterative algorithm [24] requires solving the Hungarian problem for $N = \{1, 2, 3, 4\}$.

VI. Conclusions

In this paper, we considered the spatio-temporal multi-task assignment problem for playing piano music using a team of robots. The dynamic resource allocation with multi-task assignments (DREAM) approach can be directly used to compute the minimum team size and the optimal assignments (which minimizes the total distance traveled). The DREAM approach solves the bottleneck issue of iterative computation for the required number of robots and provides the two-step solution to compute the required minimum number of robots and their optimal assignments to execute given spatio-temporal tasks. This paper analyses the DREAM algorithm and proves that the solution gives collision-free trajectories for all robots when operated in convex regions. The working of the DREAM approach is demonstrated for the piano music-playing robot simulations in a ROS2-Gazebo environment. Future work focuses on the heterogeneous robots playing music on more complex and various instruments and the exploitation of guaranteed collision-free trajectories for various STMTA problems.

Acknowledgement

The authors would like to acknowledge the financial support provided by Nokia CSR funds.

References

[1] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, and P. J. Stuckey, "Integrated task assignment and path planning for capacitated multi-agent pickup and delivery," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5816–5823, 2021.

[2] R. Bardhan, T. Bera, and S. Sundaram, "A decentralized game theoretic approach for team formation and task assignment by autonomous unmanned aerial vehicles," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 432–437.

[3] G. Notomista, S. Mayya, S. Hutchinson, and M. Egerstedt, "An optimal task allocation strategy for heterogeneous multi-robot systems," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 2071–2076.

[4] K. Harikumar, J. Senthilnath, and S. Sundaram, "Mission aware motion planning (map) framework with physical and geographical constraints for a swarm of mobile stations," *IEEE transactions on cybernetics*, vol. 50, no. 3, pp. 1209–1219, 2019.

[5] T. Venugopalan, K. Subramanian, and S. Sundaram, "Multi-uav task allocation: A team-based approach," in *2015 IEEE Symposium Series on Computational Intelligence*. IEEE, 2015, pp. 45–50.

[6] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," *Cooperative robots and sensor networks 2015*, pp. 31–51, 2015.

[7] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495–1512, 2013.

[8] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification," *Operational research*, vol. 22, no. 3, pp. 2033–2062, 2022.

[9] O. Cheikhrouhou and I. Khoufi, "A comprehensive survey on the multiple traveling salesman problem: Applications, approaches and taxonomy," *Computer Science Review*, vol. 40, p. 100369, 2021.

[10] D.-H. Lee, "Resource-based task allocation for multi-robot systems," *Robotics and Autonomous Systems*, vol. 103, pp. 151–161, 2018.

[11] Y. Xu, T. Zhang, J. Loo, D. Yang, and L. Xiao, "Completion time minimization for uav-assisted mobile-edge computing systems," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12 253–12 259, 2021.

[12] H. Luan, Y. Xu, D. Liu, Z. Du, H. Qian, X. Liu, and X. Tong, "Energy efficient task cooperation for multi-uav networks: A coalition formation game approach," *IEEE Access*, vol. 8, pp. 149 372–149 384, 2020.

[13] E. Nunes, M. Manner, H. Mitiche, and M. Gini, "A taxonomy for task allocation problems with temporal and ordering constraints," *Robotics and Autonomous Systems*, vol. 90, pp. 55–70, 2017.

[14] B. Niu, F. Zeng, and L. Chen, "Impact of promised-delivery-time and brand image on imported vaccine provider's agency marketing strategy," *Computers & Industrial Engineering*, vol. 162, p. 107748, 2021.

[15] H. Cañas, J. Mula, F. Campuzano-Bolarín, and R. Poler, "A conceptual framework for smart production planning and control in industry 4.0," *Computers & Industrial Engineering*, vol. 173, p. 108659, 2022.

[16] Y. Abdelsadek and I. Kacem, "Productivity improvement based on a decision support tool for optimization of constrained delivery problem with time windows," *Computers & Industrial Engineering*, vol. 165, p. 107876, 2022.

[17] T. Cokyasar, A. Subramanyam, J. Larson, M. Stinson, and O. Sahin, "Time-constrained capacitated vehicle routing problem in urban e-commerce delivery," *Transportation Research Record*, vol. 2677, no. 2, pp. 190–203, 2023.

[18] Y. Yin, D. Li, D. Wang, J. Ignatius, T. Cheng, and S. Wang, "A branch-and-price-and-cut algorithm for the truck-based drone delivery routing problem with time windows," *European Journal of Operational Research*, 2023.

[19] Z. Miao, A. Lim, and H. Ma, "Truck dock assignment problem with operational time constraint within crossdocks," *European journal of operational research*, vol. 192, no. 1, pp. 105–115, 2009.

[20] M. Maknoon, O. Koné, and P. Baptiste, "A sequential priority-based heuristic for scheduling material handling in a satellite cross-dock," *Computers & Industrial Engineering*, vol. 72, pp. 43–49, 2014.

[21] J. Rupp, N. Boysen, and D. Briskorn, "Optimizing consolidation processes in hubs: The hub-arrival-departure problem," *European Journal of Operational Research*, vol. 298, no. 3, pp. 1051–1066, 2022.

[22] S. Velhal, S. Sundaram, and N. Sundararajan, "Dynamic resource allocation with decentralized multi-task assignment approach for perimeter defense problem," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 4, pp. 3313–3325, 2022.

[23] S. Chopra and M. Egerstedt, "Multi-robot routing under connectivity constraints," *IFAC Proceedings Volumes*, vol. 45, no. 26, pp. 67–72, 2012.

[24] ——, "Heterogeneous multi-robot routing," in *2014 American Control Conference*. IEEE, 2014, pp. 5390–5395.

[25] ——, "Spatio-temporal multi-robot routing," *Automatica*, vol. 60, pp. 173–181, 2015.

[26] Y. Zhang and L. E. Parker, "Multi-robot task scheduling," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 2992–2998.

[27] E. Bischoff, F. Meyer, J. Inga, and S. Hohmann, "Multi-robot task allocation and scheduling considering cooperative tasks and precedence constraints," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 3949–3956.

[28] Y. Monden, *Toyota production system: an integrated approach to just-in-time*. CRc Press, 2011.

[29] S. Emde and N. Boysen, "Optimally routing and scheduling tow trains for jit-supply of mixed-model assembly lines," *European Journal of Operational Research*, vol. 217, no. 2, pp. 287–299, 2012.

[30] S. Velhal, S. Sundaram, and N. Sundararajan, "A decentralized multirobot spatiotemporal multitask assignment approach for perimeter defense," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3085 – 3096, 2022.

[31] S. Velhal, S. B R, M. Bharatheesha, and S. Sundaram, "A dynamic heterogeneous team-based non-iterative approach for online pick-up and just-in-time delivery problems," *arXiv preprint arXiv:2304.07124*, 2023.

[32] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics (NRL)*, vol. 52, no. 1, pp. 7–21, 2005.

[33] M. Turpin, N. Michael, and V. Kumar, "Capt: Concurrent assignment and planning of trajectories for multiple robots," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 98–112, 2014.