Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 133 (2018) 676–683

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Robotics and Smart Manufacturing (RoSMa2018)

# Coordinated Selection and Timing of Multiple Trajectories of Discretely Mobile Robots

Keerthi Sagar[a,*], Jesus H. Lugo[a], Rezia Molfino[a], Dimiter Zlatanov[a], Matteo Zoppi[a], Sreekumar Muthuswamy[b]

[a] *PMAR Robotics Group, University of Genoa, Genoa 16145, Italy*

[b] *Indian Institute of Information Technology, Design and Manufacturing (IIITD&M), Kancheepuram, Chennai, 600127, India*

**Abstract**

The paper addresses the multi-agent path planning (MPP) of mobile agents with multiple goals taking into consideration the kinematic constraints of each agent. The "Swing and Dock" (SaD) robotic system being discussed uses discrete locomotion, where agents swing around fixed pins and dock with their mounting legs to realize displacement from one point to another. The system was developed as a subsystem for mobile robotic fixture (SwarmItFix). Previous work dealt with MPP for SaD agents using the concept of extended temporal graph with Integer Linear Programming (ILP) based formulations. The approach discretized time into unit steps, whereas in reality, the agents are constrained by velocity limits. Hence, a real-time schedule is required to accurately plan the agent movement in a working scenario. We utilize the concept of simple temporal network and extend our ILP formulations to model the velocity kinematic constraints. The mathematical formulations are implemented and tested using a GUROBI solver. Computational results display the effectiveness of the approach.



*Keywords:* Path planning; Obstacle avoidance; graph theory; Multi-agent system

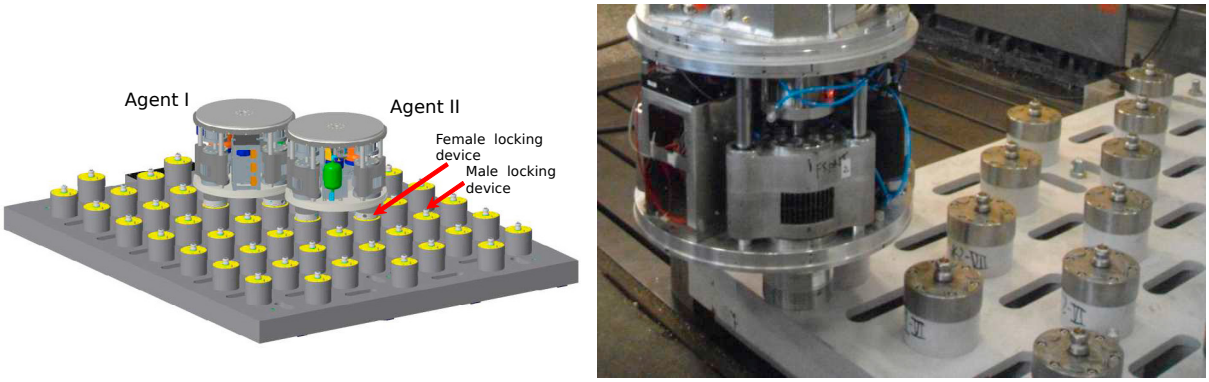
## 1. Introduction

Multi-robot path planning is a very widely studied research problem with applications in warehousing, mining, computer games and manufacturing. A special case of MPP involves planning of robots on a search graph, the discretized environment aids the use of many classic graph techniques and algorithms for producing a feasible plan.

Various techniques in Discrete MPP such as decomposing roadmaps into subgraphs (stacks and cliques) [13], network flow algorithms on graphs (roadmaps) [17], Push and Swap approach [10], M* [16], Tree based agent swapping strategy [7], and other such techniques can be found in literature. In the current article, we discuss about robotic agents with "Swing and Dock" (SaD) locomotion. The SaD locomotion was proposed for a reconfigurable fixturing agents in an aerospace application (EU FP7 SwarmItFix) [19, 8, 9]. The SaD agents swing around discrete mounting

* Corresponding author. Tel.: +39 010 353 2837
*E-mail address:* keerthi@dimec.unige.it

1877-0509 
10.1016/j.procs.2018.07.103




(a) Design (b) Prototype

Fig. 1: Swing and Dock agent

pins to realize displacements to reach adjacent positions. Hence, the SaD is inherently discrete. In previous work, we implemented a planner based on temporal graph, where the agents' centroids were abstracted to represent the nodes of a graph, and all reachable adjacent centroids with respective pins from the current centroid as edges [14]. The edges of the temporal graph were considered to be of unit weight, i.e. it takes unit time to rotate around a mounting pin and dock in adjacent pin. Integer Linear Programming (ILP) formulations are used to model the temporal graph. The planning model assumes all agents are homogeneous and execute synchronous motions. These assumptions do not translate directly to an actual operational motion plan. SaD agents are constrained by the velocity with which agent rotate, hence, we need to take into account the velocities to produce an accurate schedule and realistic time planner for the agents.

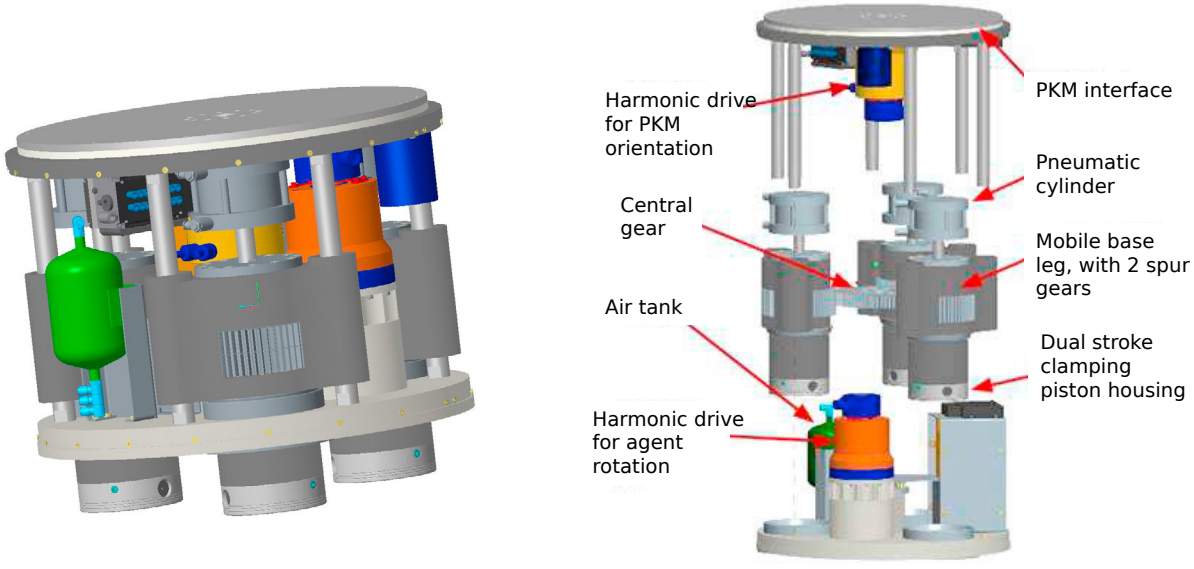
In MPP literature, many researchers have addressed the planning based on velocities of the agents as a kinematic constraint such as: generating continuous velocity profiles for multiple agents, identifying collision segments along individual paths of agents and then optimizing their velocities along the collision-free segments [11]; motion primitives to obtain kinodynamic motion planning, where pre-computed primitives capture mobility constraints of the robot and to establish a state sampling policy that is conducive to efficient search [12]; concept of Reciprocal Velocity Obstacle for real-time multi-agent navigation [1]; MAPF-POST [4], a simple temporal network to postprocess the path plan of a Multi-agent path finding solver; and many similar techniques are studied.

We adopt the methodologies proposed in [4], since before the postprocess technique, our prior ILP formulations [14] produced very similar output to the path plan generated in [4]. Hence, we extend our ILP formulations to take into account the velocity constraints of the SaD agents.

## 2. SWING AND DOCK

The SaD agents were developed to perform the role of mobile manipulators and to position themselves at discrete points on a static bench. The locomotion was invented and patented [19] for a robotic fixture project (EU FP7 SwarmItFix). A detailed description regarding the design architecture is presented in [8] and its application in the domain of material handling is discussed [15]. The discrete locomotion enables precise localization of the agents without the need of extensive sensor networks. We provide a brief description of the characteristics of the locomotion to aid the reader in comprehending the planning problem being tackled.

A fixed bench is mounted with electrical pins placed in an equilateral triangle lattice pattern, on which mobile agents can dock as shown in Fig. 1a. The fixed base (Fig. 1b) acts as a passive sub-system by housing all the electrical and pneumatic power sources. A SaD mobile agent is equipped with legs which enable the agent to dock on the mounting pins (bench). The agents are connected to the base with at least one engaged pin. The current design [9] has three mounting legs. The agents rotate around the mounting pins using a central harmonic drive as shown in Fig. 2b with a central gear and a spur gear mechanism. To translocate from one position to another, the agent needs to retract




(a) Mobile base assembly (b) Mobile base hardware

Fig. 2: SaD design elements

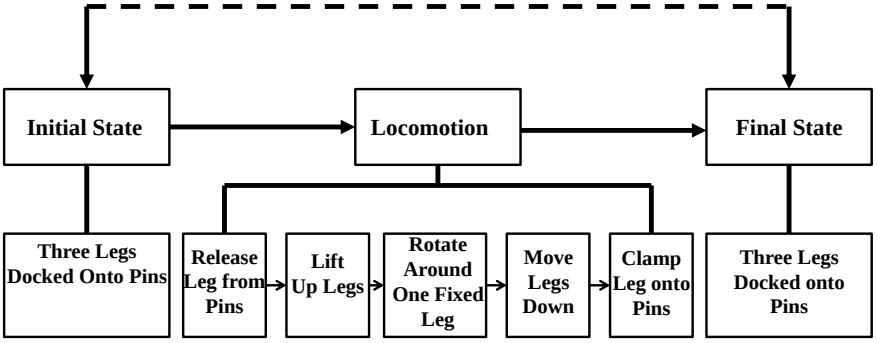



Fig. 3: Locomotion methodology [8]

two legs from the mounting pins (using pneumatic drives), and rotate around the docked pin. The leg around which rotation is made is clamped rigidly with the mounting pin.

The mounting pin, apart from acting as a clamp, provides the necessary electrical and pneumatic power to the mobile agent. Every SaD agent is equipped with hardware such as pneumatic accumulators and actuators, and on-board controllers to accurately control the mobile agents through Controller Area Networks (CAN) networks.

Agents are controlled through a central host which provides the higher level plan for all the agents, and is wirelessly communicated to the on-board computer. Hence, the mobile SaD agents operate in a cable free environment, resembling a plug and play system.

## 3. Problem Definition

Prior work dealt with multi-agent path planning with multiple goals. Agents were required to visit their assigned task goals in an ordered fashion by avoiding agent collision [14]. An extended temporal graph based apporach was used, where Integer linear programming (ILP) formulations were used to generate the planning for the agents. The objective of the formulations were to minimize the *makespan*, i.e., the total number of steps (individual rotations)

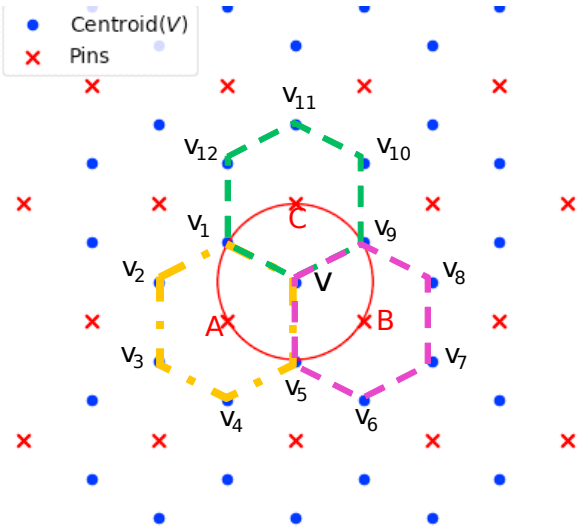



Fig. 4: Graph $G$

required until the last robot reaches its goal [18] and also to minimize the number of moves made by each agent. The ILP formulations were based on discretizing the planning time horizon, where each time step represented the amount of time required by each agent to reach its adjacent position.

Thus, the MPP problem herein involves $R \geqslant 1$ agents, each with $N_r$ labelled goals, $r = 1, \ldots, R$. We take into account the following assumptions: task allocation is already done for all agents and hence an assignment of ordered goals is available for each agent. This eases the planning as finding an optimal task allocation for the agents is NP-hard [3]. Task locations for different agents are not necessarily distinct. This constraint arises within a practical material handling scenario.

Finding even a feasible solution for the MPP is PSPACE-hard [5] even for a simplified two-dimensional case of the problem. Hence, the objective of the formulations is obtaining only a near-optimal solution. In the current article, we aim to address the kinematic constraints for the SaD agents. The kinematic constraints include provision of a maximum and minimum velocity limits for each agent to rotate around its docked pin to reach adjacent docking pins. To implement the kinematic constraints, we adapt the concept of applying a simple temporal network (STN) proposed by [4] to extend the abstract plan generated by discrete MPP to a real world schedule plan for the multi-agents.

## 4. Problem Formulation

The bench has $p$ pins in an equilateral triangular lattice with coordinates $B = \{B_1, B_2, \cdots, B_p\}$, $B_i = (x_i, y_i)$. A graph $G = (V, E)$ is defined with nodes, $V = \{v_i | i = 1, \ldots, |V|\}$, identifying the centroids of the triangles of the lattice, i.e., the possible agent positions. A connecting edge, $(v_i, v_j) \in E$, indicates the possibility to move between nodes $v_i$ and $v_j$ using only one pivot, i.e., their triangles are adjacent (share a side or a vertex). The representation of SaD agent on an equilateral triangle lattice bench is depicted in Fig. 1 & 4. Agents can reach adjacent positions by rotating $\{60°, 120°, 180°\}$ clockwise and counter-clockwise around each docked leg. Hence, a 3-legged agent can have at-most 13 adjacent reachable positions including itself as shown in Fig. 4. All edges are considered to be of unit weight, this is a valid assumption as undocking, rotating and docking are considered as an unit valid step in the graph $G$.

The adjacency list of the vertex $v$ is $\text{Adj}(v) \subseteq V$, where $v \in V$ and $w \in \text{Adj}(v)$ iff connecting edge $e = \{(v, w)\} \in E$. Some adjacent nodes have two pins that can be used as pivots. One pin is selected at random during the decision to pivot. The initial locations of SaD agents is an injective map $a_I : \{1, \cdots, R\} \rightarrow V$, where $R < |V|$. Multiple task goals are assumed for each agent. The network is based on a fixed discrete time horizon, $T$, creating $T$ copies of the vertices of $G$ [18]. Fig. 5 shows a representation of the temporal graph. To achieve sequential visiting of destinations, the goal set should be a unique list of nodes. The ordered list of tasks to be visited by agent $r$ is $W_r = \{w_1, w_2, \cdots, w_{N_r}\}$, $w_i \in V$.

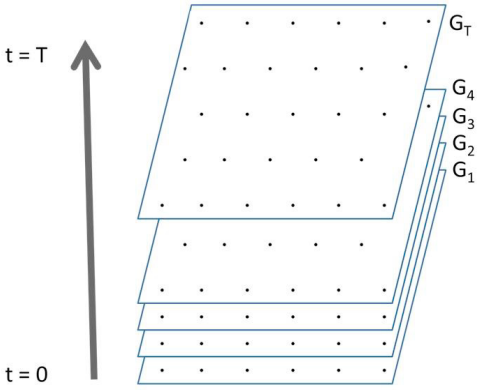



Fig. 5: Temporal graph

Obstacles are modeled as convex shaped objects and with a point in polygon test [6], we identify the nodes/centroids inside the obstacles.

Two ILP formulations based on edges of a graph and vertices of a graph were proposed in [14] to solve the path planning for the SaD multi-agents. It was observed that a vertex based formulation resulted in faster computation and fewer failures in producing a feasible solution. In the vertex based formulations, $x_{i,t,r}$ binary variables are created with size of $O(|V|TR)$, where $V$ is the number of centroids, $T$ is the time horizon and $R$ is the number of SaD agents in the planning graph. Binary decision variable $x_{i,t,r} = 1$, indicates the presence of agent $r$ in node/centroid $i$ at time step $t$. The objective function is to minimize the *makespan*, where *makespan* is the number of steps required until last agent reaches its goal. Based on [14], the objective function is defined by the Eq. 1.

$$\text{Minimize} \quad f_1 = \sum_{i \in V} \sum_{t=0}^{T} x_{i,t,r} \tag{1}$$

$$\text{for some arbitrary agent } r,\ 1 \leqslant r \leqslant R$$

We generate the near optimal solution for the path planning agents. The path plan generated for each agents $r$ is enlisted in $\delta_r$, where $\delta_r = (i, j, \cdots, k)$, $x_{\delta_r(t),t,r} = 1, 1 \leq r \leq R, 0 \leq t \leq T$. Following the methodologies developed in [4] we construct the Temporal Plan graph (TPG).

We model the constraint equations to incorporate the velocities for the agents, we can take into account for each agent $r$ both its maximum velocity $v_{\max}(r)$ and minimum $v_{\min}(r)$ rotational velocity. Since all agents were considered to be point objects in [4], all vertices in TPG were elements of the original discretized graph. Whereas, in our scenario, this assumption may lead to collision when two distinct centroids share the same mounting pin. Two different TPG were constructed for our problem, one based on adjacent vertices and other based on mounting pins. TPG based on adjacent vertices: we have a bijective map, Adj:$\delta_r \leftrightarrow \delta'_r$, and the other based on mounting pins: $f : v \leftrightarrow \delta'_r$, for some $v \in V$ and $\delta'_r$ is a tuple representing the unique list of mounting pins for each centroid. Hence, we model a modified TPG, where $G' = (V', E')$ , where $v' \in V'$ and $v' \in \delta'_r$.

Two edges are defined: Edge type I indicating movement of agents from one node to another $((\delta'_r(t), \delta'_r(t+1)) \in E')$ and edge type II indicating precedence of agents to occupy specific nodes $((\delta'_{r'}(t') \cap \delta'_r(t)) \neq \emptyset \rightarrow (\delta_r(t)', \delta_{r'}(t'))' \in E'$, where $:r \neq r', t \in [0, \text{len}(\delta_r)], t' \in [t+1, \text{len}(\delta'_r)], r \in [1, R], r' \in [1, R]$).

An illustrative example is shown in Fig. 6, where two agents move from their initial state to their goal state. Fig. 6 depicts the respective position of the agents and their occupied mounting pins on the graph $G$. Table. 1 shows the centroids occupied by each agent during each time step. The TPG for this particular scenario is depicted in Fig. 7, where the centroids are inside the red circle and their corresponding mounting pins are inside the bigger black circle.

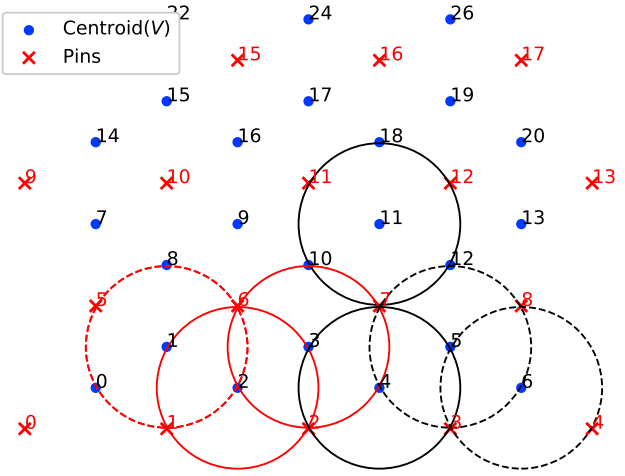



Fig. 6: Illustrative example of SaD agents' movement. Red circle represents Agent I, black circle represents Agent II, dotted circle represents intermediate positions of the agents, and solid circle represents initial and goal positions. Centroids and pins are numbered for depicting the nodes in a TPG for this example

Table 1: Agent time plan sequence for illustrative example

| | Initial | Goal | Sequence |
|---|---|---|---|
| Agent I | 2 | 3 | $2 \rightarrow 1 \rightarrow 1 \rightarrow 3$ |
| Agent II | 11 | 4 | $11 \rightarrow 4 \rightarrow 5 \rightarrow 6$ |

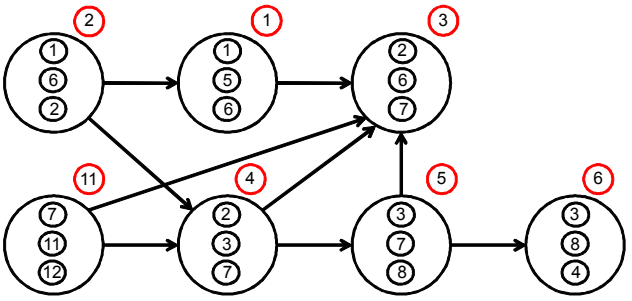


Fig. 7: Temporal Plan graph (TPG) for illustrative example

It is very clear by observation, that edge type I are the horizontal arrows among nodes and other configuration arrows depict edge type II. In Fig. 7, we establish relationship between which pins are available to be occupied by different agents, hence mapping the centroid to its corresponding mounting pins. A similar TPG constructed based on mapping centroids to their corresponding adjacency list resulted in far more edge type II. Hence, we map the centroids to their respective docking pins rather than the adjacency based list to produce a feasible plan. In the centroid graph $G$ in Fig. 4, it is observed that all edges have unit weight. In the TPG, we can model a more accurate constraint, where the centroids based on the rotations require $\alpha = \{60°, 120°, 180°\}$. Therefore, the angle each agent has to rotate to move from centroid $i$ to $j$ can be represented as $\mathrm{Ang}(i, j)$, where $i, j \in V$. Apart from rotation, the agent has to engage its pneumatically retractable legs onto the docking pins. We represent the time taken by agent $r$ for this particular action as $t_{\mathrm{retr}}(r)$. All these assumptions enables us to compute a very precise time planner for the agents.

Time assignments are made on the vertices in the TPG which satisfy constraints of the edges. Continuous variables $y_{i,r} \in R$ with size of $O(|T||R|)$ are created. During initial positions all agents are in a collision free state. The time assignment is initiated for each agent as: $y_{0,r} = 0, 1 \leq r \leq R$.

The temporal plan graph is then converted into a simple temporal network (STN) based on the following constraints on the TPG edges:

$$y_{\delta'_r(t+1),r} - y_{\delta'_r(t),r} \geq \text{Ang}(\delta'_r(t+1), \delta'_r(t))/v_{\min}(r) + t_{\text{retr}}(r) \tag{2}$$

$$0 \leq t \leq \text{len}(\delta_r), 1 \leq r \leq R$$

$$y_{\delta'_r(t+1),r} - y_{\delta'_r(t),r} \leq \text{Ang}(\delta'_r(t+1), \delta'_r(t))/v_{\max}(r) \tag{3}$$

$$0 \leq t \leq \text{len}(\delta_r), 1 \leq r \leq R$$

$$y_{\delta'_{r'}(t'),r'} - y_{\delta'_r(t),r} \geq \pi/(v_{\min}(r)) + t_{\text{retr}}(r) \quad \forall r' \neq r,\ (\delta'_{r'}(t') \cap \delta'_r(t)) \neq \emptyset \tag{4}$$

$$0 \leq t \leq \text{len}(\delta_r), t+1 \leq t' \leq \text{len}(\delta_{r'}), 1 \leq r \leq R, 1 \leq r' \leq R$$

Initial starting position of all agents are collision free. Number of idle moves have already been maximized by declaring a separate objective function for the ILP formulations. We test the feasibility of the formulations via simulation experiments in the following section.

## 5. SIMULATIONS

Simulations are performed on an Intel Xeon Dual Core Processor 2.0 Ghz 64-bit OS with 16.0 GB RAM, Windows 10. Mathematical models are implemented on Gurobi 8.0.0 with Python 2.7 as interface.

There are 52 pins in the actual SaD bench in the SwarmItFix prototype. To have a symmetrical grid for computation, 36 pins are considered, where the graph $G$ has $|V| = 36$ vertices and $|E| = 461$ edges. For the given graph $G$, the centroid graph $G'$ has $|V'| = 49$ and $|E'| = 491$ edges. Similar to the previous work, we generate the path plan for SaD agents with the following combination: 5 agents and 1 goal state each, 4 agents and 2 goal states each, 3 agents and 3 goal states each, and 2 agents and 4 goal states each. Agents are randomly initiated on the graphs with random obstacles. Connectivity between initial and goal states is verified with the All Pair Shortest Path (APSP) matrix. If the nodes are disconnected, the agents and obstacles are re-initiated until a connected graph is obtained. Makespan and total distance moved by each agent is minimized with the formulations. By minimizing the total distance, we increase the idling of agents in a particular configuration. Agent idling in the same configuration leads to lesser number of vertices in the TPG graph.

Based on the formulation described in the earlier section we create the TPG and STN. The STN is then converted to an equivalent distance graph. We detect negative cycles in the distance graph, since absence of negative cost cycles in the distance graph is equivalent to the existence of a schedule that satisfies all simple temporal constraints [2].

$$\text{Minimize} \quad f_2 = \sum_{r=1}^{R} \sum_{t=0}^{\text{len}(\delta_r)} y_{\delta'_r(t),r} \tag{5}$$

We detect negative cycles in the graph using Bellman-Ford algorithm. The constraints are implemented in GUROBI solver. Similar to our prior work, we minimize the makespan time for the STN using Eq. 5. The constraints defined in the previous section aids to convert the STN to a Linear Programming problem.

Since, the number of agents for a practical working scenario is small and the time horizon is in the size of 10 time steps to 15 time steps, the time plan converged to a feasible solution on an average of 15-30 seconds for the experiments. This observation also suggests that for a bigger bench size, the computation time required may not exceed a few minutes. The plan obtained can be conveniently transfered to the controller for real time execution of the SaD agents.

## 6. Conclusion and Future Work

A multi-agent system with Swing and Dock locomotion required a more efficient time planner to be applicable with the hardware systems. We extend the discrete path plan generated by the Integer Linear Programming formulations to include velocity constraints of the agents. We utilize the concept of Simple Temporal Network to model the required constraints. We minimize the makespan of all agents before they reach their respective goals. We run simulation experiments in an ILP solver to test the effectiveness of the formulations. We observed the ILP solver was able to produce optimal solutions within seconds of initiation. These experiments can avoid expensive reformulation of the path plan, when feasible planning is not obtained during execution of the SaD agents in the real world scenario. Experimental results of the implementation of the formulations on the test workbench will be reported in future communications.

## References

[1] Van den Berg, J., Lin, M., Manocha, D., 2008. Reciprocal velocity obstacles for real-time multi-agent navigation, in: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, IEEE. pp. 1928–1935.
[2] Dechter, R., Meiri, I., Pearl, J., 1991. Temporal constraint networks. Artificial intelligence 49, 61–95.
[3] Gao, P.a., Cai, Z.x., 2006. Multi-robot task allocation for exploration. Journal of Central South University of Technology 13, 548–551.
[4] Hönig, W., Kumar, T.S., Cohen, L., Ma, H., Xu, H., Ayanian, N., Koenig, S., 2016. Multi-agent path finding with kinematic constraints., in: ICAPS, pp. 477–485.
[5] Hopcroft, J.E., Schwartz, J.T., Sharir, M., 1984. On the complexity of motion planning for multiple independent objects; pspace-hardness of the" warehouseman's problem". The International Journal of Robotics Research 3, 76–88.
[6] Hormann, K., Agathos, A., 2001. The point in polygon problem for arbitrary polygons. Computational Geometry 20, 131–144.
[7] Khorshid, M.M., Holte, R.C., Sturtevant, N.R., 2011. A polynomial-time algorithm for non-optimal multi-agent pathfinding, in: Fourth Annual Symposium on Combinatorial Search.
[8] de Leonardo, L., Zlatanov, D., Zoppi, M., Molfino, R., 2013a. Design of the locomotion and docking system of the swarmitfix mobile fixture agent. Procedia Engineering 64, 1416–1425.
[9] de Leonardo, L., Zoppi, M., Xiong, L., Zlatanov, D., Molfino, R.M., 2013b. Swarmitfix: a multi-robot-based reconfigurable fixture. Industrial Robot: An International Journal 40, 320–328.
[10] Luna, R., Bekris, K.E., 2011. Efficient and complete centralized multi-robot path planning, in: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE. pp. 3268–3275.
[11] Peng, J., Akella, S., 2005. Coordinating multiple robots with kinodynamic constraints along specified paths. The International Journal of Robotics Research 24, 295–310.
[12] Pivtoraiko, M., Kelly, A., 2011. Kinodynamic motion planning with state lattice motion primitives, in: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE. pp. 2172–2179.
[13] Ryan, M., 2006. Multi-robot path planning with sub-graphs, in: Proceedings of the 19th Australasian Conference on Robotics and Automation, Auckland, New Zeland, Australian Robotics & Automation Association, Citeseer.
[14] Sagar, K., Zlatanov, D., Zoppi, M., Nattero, C., Muthuswamy, S., 2017. Multi-goal path planning for robotic agents with discrete-step locomotion, in: ASME 2017 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers. pp. V05AT08A033–V05AT08A033.
[15] Sagar, K., Zlatanov, D., Zoppi, M., Nattero, C., Sreekumar, M., 2016. Path planning of a material handling agent with novel locomotion, in: Proceedings of the ASME IDETC, pp. V004T05A007–V004T05A007.
[16] Wagner, G., Choset, H., 2011. M*: A complete multirobot path planning algorithm with performance bounds, in: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE. pp. 3260–3267.
[17] Yu, J., LaValle, S.M., 2013a. Multi-agent path planning and network flow, in: Algorithmic Foundations of Robotics X. Springer, pp. 157–173.
[18] Yu, J., LaValle, S.M., 2013b. Planning optimal paths for multiple robots on graphs, in: IEEE ICRA, pp. 3612–3617.
[19] Zoppi, M., Molfino, R., Zlatanov, D., 2013. Bench and method for the support and manufacturing of parts with complex geometry. US Patent 8,495,811.