Multi-Robot Layout Optimization for Efficient Cyclic Production Lines

Bo Ying, Su[1] and Changliu Liu[1]

***Abstract*— This study introduces the Multi-Robot Layout Optimization (MRLO) framework, designed to enhance efficiency in cyclic multi-robot production lines by optimizing robot layout to minimize cycle times and energy consumption. Utilizing Petri nets for modeling the production workflow's temporal dynamics, MRLO leverages the Slack Convex Feasible Set (SCFS) algorithm for determining time-optimal and energy-efficient robot trajectories and layout. Subsequently, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is employed for global optimization, focusing on the improvement of cycle times and energy use. Simulation results validate the framework's effectiveness, demonstrating substantial efficiency gains in cyclic multi-robot production environments.**

I. INTRODUCTION

The design and optimization of robot layouts in production lines have become a critical area of focus within the broader field of robotics, emphasizing the strategic arrangement of multi-robot systems for enhanced operational safety and efficiency. *Safety* aims to prevent collisions during robot operations, and *efficiency* involves minimizing cycle time while reducing energy consumption. Various strategies for layout optimization have been proposed, including evolutionary algorithms, as seen in [1] and [2], and gradient-based position optimization methods, such as [3] [4]. Furthermore, optimization of warehouse layout has been explored by [5], which discretizes the environment and employs evolutionary algorithms to maximize throughput of multi-agent operations.

Existing layout optimization methods tend to focus on either the robots' manipulability [3] or specific robot endeffector poses [6] without accounting for task-level coordination and workflow constraints, or they tackle the broader issue of multi-robot coordination for layout but using static trajectory optimization [5]. This approach overlooks the solution space of possible robot trajectories within the workflow. Our work bridges this gap by integrating robot cell layout with robotic trajectory optimization, considering both the detailed dynamics of individual robots and the scheduling workflow.

Workflow scheduling involve setting the sequence and timing of the tasks to ensure a smooth and efficient process. Robot dynamics focus on planning the movement of individual robots to ensure their paths are free from collisions and optimized in terms of time. Integrating both is vital for effective robot layout optimization that minimizes energy use without compromising on cycle times.

This paper introduces the Multi-Robot Layout Optimization (MRLO) framework, illustrated in Fig. 1. MRLO adopts a bilevel optimization approach. At the upper level, global optimization aims to identify the best robot layout within the production setting to optimize cycle times and energy usage. The lower level focuses on achieving a locally optimal layout that ensures time-efficient, collision-free, and energy-saving robot trajectories. Initially, the global optimization phase uses an evolutionary algorithm to establish preliminary robot layout, which are then refined at the lower level. Here, the refinement starts with determining layout and trajectories that are time-optimized, followed by a workflow scheduling process that seeks execution schedule yielding the shortest cycle times. The process concludes with an energy optimization phase, which adjusts the trajectories to reduce energy consumption without extending the cycle time. The solution from the lower level is then fed back to the global optimization phase, and the process iterates until convergence is achieved.

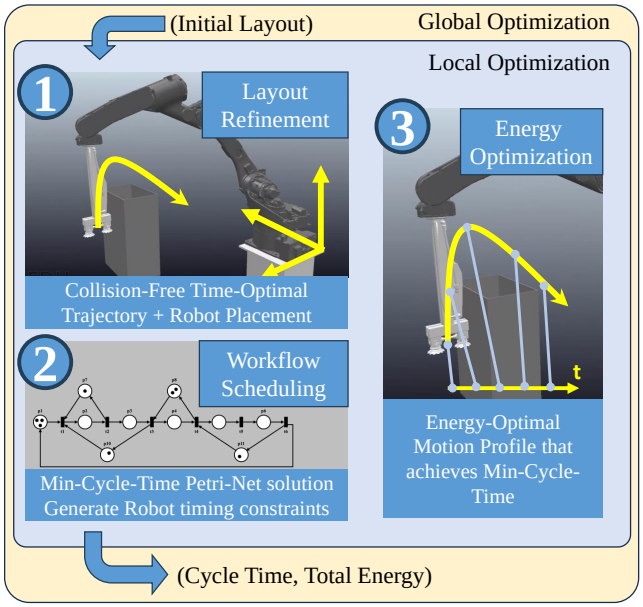


Fig. 1: The Multi-Robot Layout Optimization Framework (MRLO) runs in a loop with global optimization and local optimization. The global optimization phase uses an evolutionary algorithm to establish preliminary robot layout, which are then refined at the lower level. The local optimization result is used as the objective value that is then fed back to the global optimization.

Given the evolutionary algorithm's need for numerous samples to achieve convergence, we suggest a fast and deterministic method for evaluating system performance during local optimization. Rather than addressing the direct collo-

[1]Carnegie Mellon University, Pittsburgh, PA. Contact: {boyings, cliu6}@andrew.cmu.edu

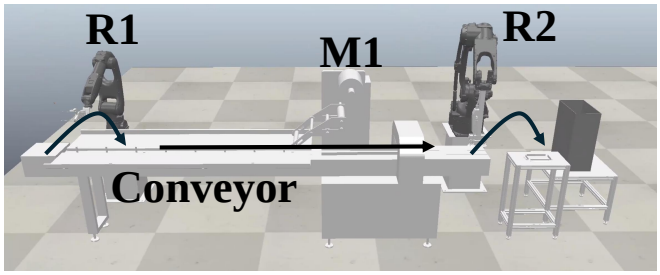


Fig. 2: A Two-Robot Sequential pipeline with one conveyor. The pipeline is to transfer workpieces from the left to the right, processed by a machine $M1$ at the end of the conveyor.

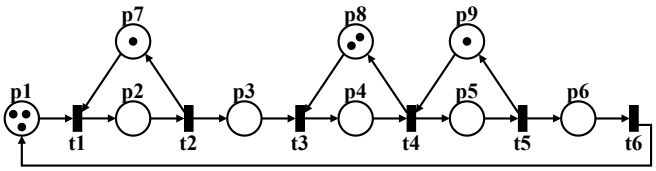
cation problem with generic non-linear optimization solvers like sequential quadratic programming (SQP) [7]—which tend to be inefficient with highly non-convex constraints—we employ the slack convex feasible set (SCFS) algorithm [8], which greatly speeds up solving the trajectory through convexification. By applying SCFS to the layout optimization problem, we efficiently find the pair of near-optimal robot layout and trajectory that are both collision-free and time-optimized within just a few iterations. The SCFS algorithm is similarly effective in the energy optimization phase, where it swiftly adjusts robots' motion profiles to lower energy use.

The contributions of this paper are as follows: 1) We introduce local optimization of robot layout which greatly speeds up the global optimization process, 2) Through Workflow Scheduling, we explore the solution space of energy efficient robot trajectories for a given layout such that the optimal cycle time is preserved, and 3) We demonstrate the effectiveness of the MRLO framework through simulations, showing substantial efficiency gains in cyclic multi-robot production environments. The rest of this paper is organized as follows. Section II formulates the multi-robot layout optimization problem, Section III introduces the multi-robot layout optimization (MRLO) framework, Section IV evaluates the performance of the MRLO framework experimentally through simulations. Section V concludes the paper.

II. PROBLEM FORMULATION

A. Multi-Robot Production Line

Consider a cyclic production line featuring R robots. As illustrated in Figure 2, a typical setup might include a two-robot production line with a single conveyor. In this scenario, robot R_1 is tasked with transferring incoming workpieces to the conveyor belt. These workpieces are then moved rightward by the conveyor to a machine at the conveyor's end for processing, after which robot R_2 picks them up. Robot R_2 is responsible for transferring the processed workpieces to an outgoing area, where they are collected by a human operator. Within this setup, the movements and timing of the robots are to be optimized by an algorithm, while the conveyor's motion and the human operator's tasks are predetermined. The production line also contains various environmental obstacles, including machines and the conveyor itself, necessitating that the robots navigate through these



Place	Description		
p1	Workpiece Available to be picked up by Robot 1		
p2	Workpiece and Robot 1 Ready		
p3	Workpiece on conveyor		
p4	Workpiece ready to be picked up by Robot 2		
p5	Workpiece and Robot 2 Ready		
p6	Workpiece arrived at end of the pipeline		
p7	Robot 1 done unloading workpiece to conveyor		
p8	Conveyor slot available (# free represented as tokens)		
p9	Robot 2 done unloading workpiece to final location		
Transition	**Description**	**Time Delay**	**Energy**
t1	Robot 1 moves to pick pose	variable	variable
t2	Robot 1 moves workpiece to conveyor	variable	variable
t3	Conveyor moves, and workpiece processed	10	5
t4	Robot 2 moves to pick pose	variable	variable
t5	Robot 2 moves workpiece to conveyor	variable	variable
t6	Workpiece removed by human worker	20	0

Fig. 3: The Petri Net Model for the A Two-Robot production line. The pipeline consists of two robots and one conveyor, where the robot motion timings are variable and the conveyor motion is fixed.

without causing collisions. The objective is to determine the most efficient robot layout that minimize both cycle time and energy consumption.

B. Timed Petri Net Model for Cyclic Production Line

We model the operation of a cyclic production line using Timed Petri net, with an extension of the energy function. Figure 3 shows the Petri Net model of the Two-Robot production line mentioned in the previous section. A Timed Petri Net with energy is a 7 tuple $G = (P, T, I, O, \mathcal{M}_0, \boldsymbol{\tau}, \boldsymbol{e})$, where P is the set of places, shown as circles, T is the set of transitions, shown as rectangle bars with arrows pointing in and out, I is the input function, O is the output function, $\mathcal{M}_0$ is the initial marking, and $\boldsymbol{\tau}$ is the timing vector. The marking $M(p)$ is a vector of non-negative integers representing the number of tokens in place p, shown as the dots in the places. The input function is defined for each transition $I : P \times T \to \mathbb{N}$, where $I(p,t)$ is the number of tokens consumed from place p when transition t fires. Similarly, the output function is defined as $O : P \times T \to \mathbb{N}$, where $O(p,t)$ is the number of tokens produced in place p when transition t fires. The energy vector $\boldsymbol{e} \in \mathbb{R}^{|T|}$ relates each transition t to a scaler quantity of energy consumption. The timing vector $\boldsymbol{\tau} \in \mathbb{R}^{|T|}$ is the time required for transition t to fire. A transition t is *enabled* if there are enough tokens in the input places. That is, $M(p) \geq I(p,t)$ for any place p. A transition t can fire at time i if there have been $M(p) \geq I(p,t)$ for the time interval $[i - \boldsymbol{\tau}_t, i]$. After transition t is fired, the marking

of the Petri net evolves according to the following equation:

$$M_i(p) = M_{i-\boldsymbol{\tau}_t}(p) + O(p,t) - I(p,t) \tag{1}$$

where $O(p,t)$ is the number of tokens produced in place p when transition t fires, and $I(p,t)$ is the number of tokens consumed from place p when transition t fires. In the Figure 3 example, $I(p,t) = 1$ if the transition t consumes a token from place p (there is an arrow pointing from place p to transition t), and $O(p,t) = 1$ if the transition t produces a token in place p (there is an arrow pointing from transition t to place p). $I(p,t) = 0$ and $O(p,t) = 0$ otherwise.

A solution to a Petri Net G over a time horizon k is given as a firing schedule $\mathbf{s} = [s_1^T, s_2^T, \ldots, s_k^T] \in \mathbb{1}^{|T|\times k}$ where the binary vector s_i denotes the tasks firing at timestep i. The duration vector $\boldsymbol{\delta} = [\delta_1, \delta_2, \cdots, \delta_k] \in \mathbb{R}^k$ captures the time intervals between consecutive timesteps, with δ_i specifying the duration from timestep i to $i+1$.

The cycle time, the sum of all durations within $\boldsymbol{\delta}$, represents the total time required for a complete cycle of tasks. energy consumption is quantified as the sum $\max_{i\in[1,k]} e(t)^T \cdot s_i$, where $e(t)$ is the energy function.

C. Robot Layout and Motion Planning

For each robot, we find the optimal base placement and trajectory for 1) time optimal and 2) energy optimal objectives, subject to robot dynamics and collision avoidance constraints. The Robot Layout and Motion Planning problem is formulated as follows: Let the robot's state be represented by $x \in \mathcal{X} \subseteq \mathbb{R}^n$, with $\mathcal{X}$ indicating the feasible state space in n dimensions. The position of the robot base is defined as $b \in \mathcal{B} \subseteq \mathbb{R}^d$, with $\mathcal{B}$ encompassing all potential reference base positions, assumed to remain fixed throughout the production cycle. The task specification for the robot is $w \in W \subseteq T$, and the robot's discrete-time path is $x^w = \begin{bmatrix} x_1^{w^T} & x_2^{w^T} & \ldots & x_{N^w}^{w^T} \end{bmatrix}^T \in \mathcal{X}^{N^w}$, where $x_i^w \in \mathcal{X}$ is the state at time step i, and N^w is the time horizon of task w. The robot's time profile is $t^w = \begin{bmatrix} t_1^w & t_2^w & \ldots & t_{N^w-1}^w \end{bmatrix}^T \in \mathbb{R}^{N^w-1}$, defined as the time intervals between consecutive time step i and $i+1$.

1) Joint and Cartesian Constraints: The motion of the robot R_i is subject to the following joint constraints for each task w:

$$x_i^w \in \mathcal{X}, \quad \forall i = 1, \ldots, N^w \tag{2}$$
$$x_i^w = x_i^r, \forall i = 1, \ldots, J^w \tag{3}$$

where J^w is the number of joint constraints for task w. The motion of the robot R_i is subject to the following Cartesian constraints at its endeffector for each task w:

$$FK(R_i, x_i^w, b) = p_i^w, \quad \forall i = 1, \ldots, C^w \tag{4}$$

where C^w is the number of Cartesian constraints for task w, FK is the forward kinematics function, and p_i^w is the endeffector position at time step i for task w.

2) Obstacle Avoidance Constraints: The Cartesian space occupied by the robot at state x_i^w, given the fixed base position b, is $\mathcal{C}(x_i^w, b) \in \mathbb{R}^3$. The environment's obstacles at time t occupy $\mathcal{O}_i \in \mathbb{R}^3$. The minimum distance between the robot and obstacles, with the base position b, is as follows:

$$d(x_i^w, b, \mathcal{O}_i) := \min_{p_R \in \mathcal{C}(x_i^w, b), p_O \in \mathcal{O}_i} d_E(p_R, p_O) \tag{5}$$

where $d_E(p_R, p_O)$ is the Euclidean distance between points p_R and p_O. The feasible trajectory of the robot is subject to the following obstacle avoidance constraints:

$$d(x_i^w, b, \mathcal{O}_i) \geq d_{min}, \quad \forall i = 1, \ldots, N^w \tag{6}$$

where d_{min} constrains the minimum distance between the robot and obstacles.

3) Robot Dynamics Constraints: Given a discretized robot path x^w and time profile t^w, the robot dynamics constrains the maximum velocity, acceleration and jerk of the robot, which are defined as follows:

$$v_i^w = \frac{x_{i+1}^w - x_i^w}{t_i^w} \qquad a_i^w = \frac{2(v_i^w - v_{i-1}^w)}{t_i^w + t_{i-1}^w} \tag{7}$$
$$j_i^w = \frac{3(a_i^w - a_{i-1}^w)}{t_i^w + t_{i-1}^w + t_{i-2}^w} \tag{8}$$

The robot's dynamics constraints are defined as follows:

$$v_{min}^w \leq v_i^w \leq v_{max}^w, \quad \forall i = 1, \ldots, N^w \tag{9}$$
$$a_{min}^w \leq a_i^w \leq a_{max}^w, \quad \forall i = 1, \ldots, N^w \tag{10}$$
$$j_{min}^w \leq j_i^w \leq j_{max}^w, \quad \forall i = 1, \ldots, N^w \tag{11}$$

4) Time and Energy Objectives of Robot Trajectories: The time objective of a robot trajectory is simply the sum of the time profile t^w, which is defined as follows:

$$time(x^w, t^w) = \sum_{i=0}^{N^w} t_i^w \tag{12}$$

The energy objective of a robot trajectory is defined as the sum of energy consumption over the time horizon, which is defined as follows:

$$energy(x^w, t^w) = \frac{1}{\sum_{i=0}^{t^w}} \sum_{i=0}^{N^w} \|\tau_i^w \cdot (x_{i+1}^w - x_i^w)\| \tag{13}$$

where the torque τ_i^w is defined as:

$$M^w(x_i^w)a_i^w + C^w(x_i^w, v_i^w)v_i^w + N^w(x_i^w) = \tau_i^w \tag{14}$$

and M^w is the mass matrix, C^w is the Coriolis and centrifugal matrix, and N^w is the gravity and friction matrix for the robot operating task w.

D. The Layout Optimization Problem

Given R robots and the Petri Net model of the workflow G, the layout optimization problem aims to find the optimal reference base positions $\mathbf{b}^r \in \mathbb{R}^{d\times R}$ for each robot, such that the total cycle time and energy consumption are minimized.

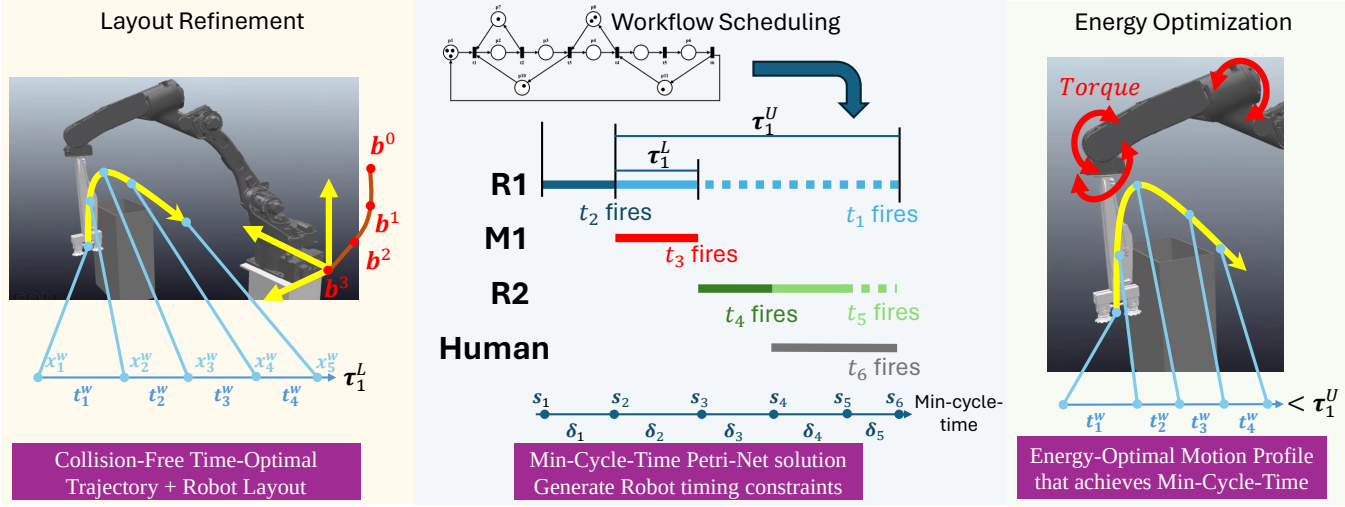


Fig. 4: A example of local optimization process on Two-Robot pipeline with conveyors. It consists of three steps: 1) Layout Temporal Optimization to find the time-optimal robot layout and trajectories 2) Workflow Scheduling to find the optimal task schedule that minimizes cycle time and extract the timing constraints $\boldsymbol{\tau}^{\boldsymbol{L}}$ and $\boldsymbol{\tau}^{\boldsymbol{U}}$ for each robot motion, and 3) energy Optimization to adjust the robot trajectories to minimize energy consumption.

d is the degree of freedoms of the robot bases. The layout optimization problem is formulated as follows:

$$\min_{\mathbf{b}^r} J_L(\mathbf{b}^r) = K(\mathbf{b}^r) + \lambda F(\mathbf{b}^r) \tag{15}$$

where the cycle time objective $K(\mathbf{b}^r)$ and energy consumption objective $F(\mathbf{b}^r)$ comes directly from a solution to the Petri Net. Note that the robots sitting at reference base position $\mathbf{b}^r$ may not be collision-free or satisfy safety margin constraints. The local optimization would jointly solve for robot layout and trajectories.

III. MULTI-ROBOT LAYOUT OPTIMIZATION FRAMEWORK

The Multi-Robot Layout Optimization (MRLO) framework is composed of the upper level global optimization and the lower level local optimization, as shown in Figure 1. The global optimization applies Covariance Matrix Adaptation Evolution Strategy (CMA-ES) to find the optimal robot layout based on the fitness value, which is obtained from optimizing cycle time and energy consumption in the lower level local optimization. The local optimization shown in Figure 4 consists of three steps: 1) Layout Temporal Optimization to find the time-optimal robot layout and trajectories 2) Workflow Scheduling to find the optimal task schedule that minimizes cycle time and extract the timing constraints for each robot motion, and 3) energy Optimization to adjust the robot trajectories to minimize energy consumption. The local optimization result is used as the objective value that is then fed back to the global optimization. The process iterates until convergence is achieved.

A. Evolutionary Strategy for Global Layout Optimization

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a stochastic optimization algorithm that is widely used for non-linear, non-convex optimization problems. It is particularly suitable for the layout optimization problem, where the objective function is non-convex and non-linear. The mathematical foundation of CMA-ES is out of the scope of this paper, and the reader is referred to [9] for more details. Here, we describe the formulation of the fitness function for the layout optimization problem, which is exactly the objective function in (15). Given a Petri Net model of the cyclic production line G with a time horizon of k and includes variable timings and energy parameters that is to determined by robot motion, assume we have a solution to the net $(\mathbf{s}, \boldsymbol{\delta}, \boldsymbol{\tau}, \boldsymbol{e})$ where $\mathbf{s} \in \mathbb{1}^{|T|\times k}$ is the firing schedule, $\boldsymbol{\delta} \in \mathbb{R}^k$ is the duration vector , $\boldsymbol{\tau}$ is timing vector and $\boldsymbol{e}$ is energy vector. The fitness function J_L is formulated as follows:

$$\begin{aligned} J_L(\mathbf{b}^r) &= K(\mathbf{b}^r) + \lambda F(\mathbf{b}^r) \\ &= \sum_{i=1}^{k} \boldsymbol{\delta}_i + \lambda \sum_{i=1}^{k} \mathbf{s}_i^T \boldsymbol{e} \end{aligned} \tag{16}$$

The solution to the Petri Net $(\mathbf{s}, \boldsymbol{\delta}, \boldsymbol{\tau}, \boldsymbol{e})$ is obtained by the local optimization. Local optimization takes an initial robot layout and determines the optimal firing schedule and duration vector along with the optimized Petri Net parameters. The fitness function is evaluated by the CMA-ES algorithm, which iteratively updates the reference base positions $\mathbf{b}^r$ to minimize the fitness function.

B. Time-Optimal Robot Layout and Trajectory

To determine the minimum achievable cycle time of the Petri Net Model, for each transition in the net that involves robot motion, we solve for refined robot layout and time-optimal collision-free trajectories as the first step of local optimization.

Suppose we are optimizing the layout and trajectories involving robot R_i for tasks $W_i \subseteq T$. We concatenate the trajectories of all tasks W_i into a single trajectory x^w and time profile t^w. The objectives and constraints does not apply

at the boundries of the task trajectories. The time-optimal robot layout and trajectory problem is formulated as follows:

$$\begin{aligned} &\min_{x^w,t^w,b} J_T(x^w,t^w,b) = \sum_{i=1}^{k} (\|t^w\|_1) \\ &\text{s.t.} \qquad\qquad\qquad\qquad t^w > 0 \\ &G_T(x^w,t^w,b) = 0 \\ &H_T(x^w,t^w,b) \leq 0 \end{aligned} \tag{17}$$

where G_T are the equality constraints including the Joint constraints (3), the Cartesian constraints (4), and the robot dynamics (8). H_T are the inequality constraints including the obstacle avoidance constraints (6) and the robot dynamics limits (11).

To solve the time-optimal robot layout and trajectory problem formulated above, we applied the Slack Convex Feasible Set (SCFS) algorithm [8], which relaxes the original optimization problem by adding slack variables to the original constraints. Specifically, we relaxed the robot dynamics equality constraints in (8) as in [10]. The relaxed problem is then solved by Algorithm 1. In each iteration, CFS constructs the convex feasible set around a reference point:

$$\begin{aligned} \mathcal{F}(\boldsymbol{x}^{\boldsymbol{w}^k}, \boldsymbol{b}^k) &= \bigcap_i \mathcal{F}_i(\boldsymbol{x}^{\boldsymbol{w}^k}, \boldsymbol{b}^k) \\ &= \{x : \phi_i(\boldsymbol{x}^{\boldsymbol{w}^k}, \boldsymbol{b}^k) + \\ &\quad \nabla\phi_i(\boldsymbol{x}^{\boldsymbol{w}^k}, \boldsymbol{b}^k) \begin{bmatrix} \boldsymbol{x}^{\boldsymbol{w}} - \boldsymbol{x}^{\boldsymbol{w}^k} \\ \boldsymbol{b} - \boldsymbol{b}^k \end{bmatrix} \geq 0\} \end{aligned} \tag{18}$$

where $\boldsymbol{x}^{\boldsymbol{w}^k}$ and $\boldsymbol{b}^k$ are the solutions from the previous iteration as shown in Figure 4, $\mathcal{F}_i(\boldsymbol{x}^{\boldsymbol{w}^k}, \boldsymbol{b}^k)$ is the convex feasible set of the ith constraint, and $\phi_i(\boldsymbol{x}^{\boldsymbol{w}^k}, \boldsymbol{b}^k)$ is the ith constraint function. The CFS algorithm is guaranteed to converge to a local optimal solution [11], and it is computationally efficient.

Algorithm 1 The Convex Feasible Set Algorithm

1: $z^{(0)} = z^r$
2: **while** $\|z^{(k+1)} - z^k\| \geq \epsilon$ **do**
3: Find the convex feasible set: $\mathcal{F}(z^{(k)}) \subseteq \mathbb{R}^e$
4: Solve QP: $z^{(k+1)} = \arg\min_{z \in \mathcal{F}(k)} J(z)$
5: $z^r \leftarrow z^{(k+1)}, k = k + 1$
6: **end while**

C. Workflow Scheduling

Given a Petri Net G with the best possible task durations $\boldsymbol{\tau}^L$ for each tasks determined by the layout temporal optimization step, the workflow scheduling step aims to solve for the optimal sequence of robot operations in a cycle and determine the timing constraints for each task such that the minimum cycle time is preserved. Workflow Scheduling is performed in three steps. First, a maximum output objective is used to find the maximum artifacts that can be produced in a cycle. Second, a minimum cycle time objective is used to find the minimum cycle time that can be achieved such that the output is the same. Third, a maximum task duration objective is used to find the maximum duration that is allowed for each tasks such that the output is the same and the minimum cycle time is maintained. The workflow scheduling problem is formulated as follows:

Let the task schedule be a matrix $S = \left[\mathbf{s}_1^T, \mathbf{s}_2^T, \ldots, \mathbf{s}_k^T\right]^T \in \mathbb{1}^{|T| \times k}$, where $\mathbf{s}_i$ is the task firing vector at timestep i, the time slice duration vector be $\boldsymbol{\delta} \in \mathbb{R}^k$, task duration vector be $\boldsymbol{\tau} \in \mathbb{R}^{|T|}$. Let t_a be an indicator task that is only fired when an artifact is produced and $\mathbf{s}_{t_a}$ be the one-hot task firing vector with only task t_a being 1. Let $\mathbf{O}_{p,t} = O(p,t)$ be the output matrix and $\mathbf{I}_{p,t} = I(p,t)$ be the input matrix. Common to all three workflow scheduling problems, the following constraints are enforced:

$$\begin{aligned} &\mathcal{Y}_1 : M_1 = M_k = \mathcal{M}_0, \\ &\mathcal{Y}_2 : M_{i+1} = M_i + \mathbf{O}\mathbf{s}_i - \mathbf{I}\mathbf{s}_i, \quad \forall i = 0, \ldots, k-1 \end{aligned} \tag{19}$$

where M_i is the marking of the Petri net at time i. Additionally, we enforce the timing constraint: Define slack variables $M_i' \leq M_i$, $\Delta_i \leq \sum_{k=j}^{i-1} \delta_k$ for all $i = 1, \ldots, k-1$ and $j = 1, \ldots, i-1$. Define $v^{t,i} \in \mathbb{R}^{i+1}$ as the valid vector for hypothetical task t at time i. For each $t \in T$ and $i = 1, \ldots, k-1$, the timing constraint is enforced as follows:

$$\mathcal{Y}_3 : \sum_{j=1}^{i+1} v_j^{t,i} \geq 1 \tag{20}$$

$$\mathcal{Y}_4 : v_{i+1}^{t,i} * \mathbf{s}_i(t) = 0 \tag{21}$$

$$\begin{aligned} &\mathcal{Y}_5 : v_j^{t,i} * (\sum_{k=j}^{i-1} \mathbf{s}_k(t)^2 + (\mathbf{s}_i(t) - 1)^2 + \\ &\sum_{k=j}^{i-1} (M_k' - \mathbf{O}_{:,t})^2 + (\Delta_i - \tau_i)^2) = 0, \forall j = 1, \ldots, i \end{aligned} \tag{22}$$

As shown in equation (20), for each possible transition t at time i, the sum of valid vector has to be greater than 1, which means either 1) the transition is not chosen (Equation 21), or 2) the transition is chosen and is enabled since some time steps j before the transition (Equation (22)). The transition has been in enabled state (input places have enough tokens) for at least τ_i time.

The first workflow scheduling problem is formulated as follows:

$$J_\alpha = \max_{\boldsymbol{\delta}, \boldsymbol{\tau}} \sum_{i=1}^{k} \mathbf{s}_i^T \mathbf{s}_{t_a} \qquad \text{s.t. } \mathcal{Y}_c \text{ is satisfied } \forall c = 1, \ldots, 5 \tag{23}$$

where the objective is to maximize the number of artifacts produced in a cycle in the time horizon.

The second workflow scheduling problem is formulated by adding the throughput constraint

$$\mathcal{Y}_6 : \sum_{i=1}^{k} \mathbf{s}_i^T \mathbf{s}_{t_a} \geq J_\alpha \tag{24}$$

$$J_\beta = \min_{\boldsymbol{\delta}, \boldsymbol{\tau}} \sum_{i=1}^{k} \delta_i \qquad \text{s.t. } \mathcal{Y}_c \text{ is satisfied } \forall c = 1, \ldots, 6 \tag{25}$$

where the objective is to minimize the cycle time while maintaining the same output as the maximum output objective.

The third workflow scheduling problem adds the cycle time constraint from the second workflow scheduling problem

$$\mathcal{Y}_7 : \sum_{i=1}^{k} \delta_i \leq J_\beta \tag{26}$$

$$J_\gamma = \max_{\boldsymbol{\delta}, \boldsymbol{\tau}} \sum_{i=1}^{k} \tau_{t_i} \qquad \text{s.t. } \mathcal{Y}_c \text{ is satisfied } \forall c = 1, \ldots, 7 \tag{27}$$

where the objective is to maximize the task durations while maintaining the same output as the maximum output objective and the minimum cycle time objective. The workflow scheduling problems are solved as Mixed Integer Non-Linear Programming (MINLP) problem. Let the solution to the third workflow scheduling problem be the maximum allowed task duration $\boldsymbol{\tau}^U \in \mathbb{R}^{|T|}$ and the solved time slice $\boldsymbol{\delta}^L \in \mathbb{R}^k$.

D. Energy Optimization

Given the upper bounds of the task durations $\boldsymbol{\tau}^U$ associated with the task schedule with minimal cycle time, the energy optimization step aims to solve for the energy-efficient robot trajectories. The energy optimization problem is formulated as follows:

$$\min_{\boldsymbol{t}^w} \sum_{i=0}^{N^w} \| \tau_i^w \cdot (x_{i+1}^w - x_i^w) \| \tag{28}$$

subject to robot dynamics constraints (11) and maximum task duration constraints $\sum_{i=0}^{N^w} t_i^w \leq \boldsymbol{\tau}^U$.

The energy optimization problem is solved using the SCFS algorithm. Note that any $\boldsymbol{\tau}^L \leq \boldsymbol{\tau}^U$ would still be feasible solution to the workflow scheduling problem as the robots can always stop earlier than the maximum allowed task duration and wait.

E. Fitness Function Evaluation

The solved cycle time $\sum_{i=1}^{k} \boldsymbol{\delta}_i^{\boldsymbol{L}}$ from the workflow scheduling step and the solved energy use $\sum_{i=1}^{k} \mathbf{s}_i^T \boldsymbol{e}$ from the energy optimization step are used to evaluate the fitness function in the upper level optimization.

[Finish this figure]

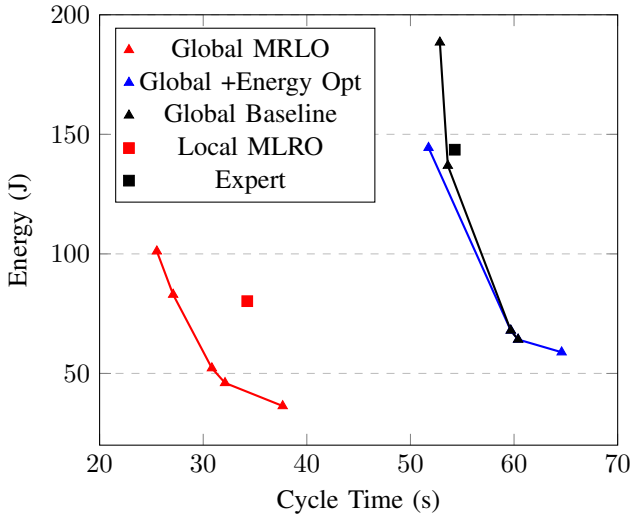


Fig. 5: Layout optimization results of Two-Robot pipeline with conveyors. The red, blue and black lines show the trade-off between energy and cycle time found by the CMA-ES global optimization by varying $\lambda = 0.0$ to 1.0 step 0.1. Even with only local optimization using Expert Layout as initial guess, significant improvement can be seen (red square) in cycle time and energy consumption over Expert Layout (black square).

IV. EXPERIMENTAL EVALUATION

In the experimental section, we evaluate the performance of the MRLO framework through simulations in 1) Two-Robot pipeline with conveyors 2) Parallel Robot Assembly Line. The performance of the MRLO framework is evaluated in terms of cycle time and energy consumption on both global and local optimization levels. The dynamical simulation is done using CoppeliaSim with MuJoCo physics engine.

A. Simulation Setup

1) Two-Robot Sequential pipeline with conveyors: The two-robot pipeline with conveyors is a cyclic production line with two robots shown in Figure 2. The robots are required to perform pick-and-place operations from the source to the conveyor belt and from the conveyor belt to the destination avoiding collision. The two robots operates sequetially, with the first robot transferring the incoming workpieces to the conveyor belt and the second robot transferring the processed workpieces to the outgoing area. The second robot has to wait for the first robot to finish and the processing is done before it can start. However, there is no such constraint for the returning motion of both robots, which presents an opportunity for energy optimization. The Petri Net model of the two-robot pipeline with conveyors is shown in Figure 3.

[Finish this figure]

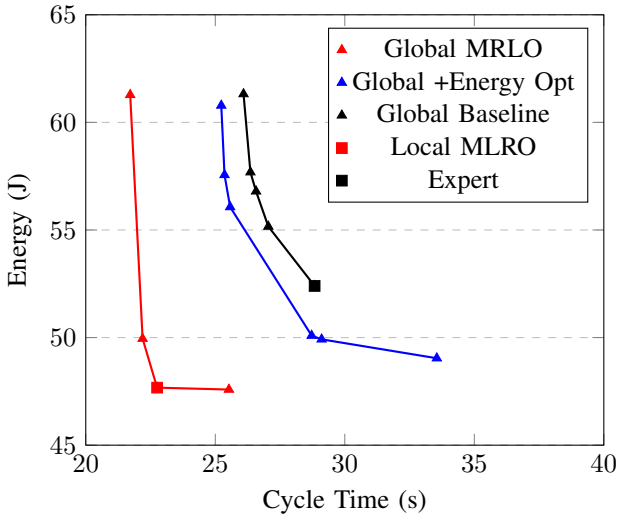


Fig. 6: Parallel Robot Assembly Line plot

[Finish this figure]

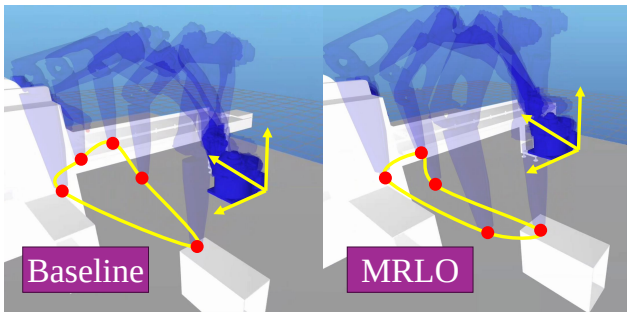


Fig. 7: Optimized GP12 Robot Layout

2) Parallel Robot Machine tending: The parallel robot assembly line is a cyclic production line with two robots is shown in Figure [missing figure]. The robots are required to perform machine tending operations. The robots operates in parallel, and so are the machines, but the cycle is only complete when both robots finished processing their workpieces and transferred to a common area. The two robots have different dynamics characteristics, and the machine processing time is different, which presents an opportunity for energy optimization when they operate in parallel. The Petri Net model of the parallel robot assembly line is shown in Figure [missing figure].

B. Results

In both production lines, we perform the following abalation study to evaluate the performance of the MRLO framework in terms of cycle time and energy consumption. The following local optimzation approaches are evaluated in the abalation study:

- **Baseline**: Performs temporal optimization as in [10].
- **Energy Optimization**: Exploit workflow schedule for reduced energy consumption.
- **Local Refinement**: Performs layout refinement in addition to energy optimization.

The three approaches are then tested with global optimization using CMA-ES. The pareto frontier in global optimization is generated by varying the weight λ in the fitness function (16).

The results of MRLO for the two-robot pipeline with conveyors and parallel robot assembly line are shown in Figure 5 and Figure 6 respectively. The results show that in local optimization, workflow and energy optimization significantly reduces the energy consumption. The local layout refinement further finds local optimal layout that reduces the cycle time. In global optimzation, we found that the MRLO framework is able to find significantly better solutions compared to the baseline with much less iterations.

C. Scalibility

Our results show that the MRLO framework with local optimization efficiently solves the layout optimization problem with much faster convergence compared to the baseline. As the SCFS algorithm usually converges in less than 10 iterations, the main bottleneck of the proposed approach is the workflow scheduling step, which solves the MINLP problem and takes around 25 seconds in both experiments.

V. CONCLUTION AND FUTURE WORK

In this paper, we proposed a Multi-Robot Layout Optimization (MRLO) framework for cyclic production lines. It combines global optimization using CMA-ES and local optimization that performs layout refinement, workflow scheduling and energy optimization. The MRLO framework is evaluated in two production lines and shows significant improvement in cycle time and energy consumption compared to the baseline. The MRLO framework is able to find significantly better solutions compared to the baseline with much less iterations. For future work, we plan to improve the computation efficiency of the workflow scheduling step to ensure scalability to more complex production lines.

VI. ACKNOWLEDGEMENT

This work was performed under the following financial assistance award 70NANB22H009 from U.S. Department of Commerce, National Institute of Standards and Technology, Advanced Functional Fabrics of America, Inc and Advanced Robotics for Manufacturing Inc. The Cad model is developed by Jack Dorman, R & D Manager of Henderson Sewing Machine Co., Inc. and refined by Chengtao Wen, Siemens.

REFERENCES

[1] Q. C. Nguyen, Y. Kim, and H. Kwon, "Optimization of layout and path planning of surgical robotic system," *International Journal of Control, Automation and Systems*, vol. 15, pp. 375–384, 2017.

[2] K. Izui, Y. Murakumo, I. Suemitsu, S. Nishiwaki, A. Noda, and T. Nagatani, "Multiobjective layout optimization of robotic cellular manufacturing systems," *Computers & Industrial Engineering*, vol. 64, no. 2, pp. 537–544, 2013.

[3] Q. Yu, G. Wang, X. Hua, S. Zhang, L. Song, J. Zhang, and K. Chen, "Base position optimization for mobile painting robot manipulators with multiple constraints," *Robotics and Computer-Integrated Manufacturing*, vol. 54, pp. 56–64, 2018.

[4] S. Mitsi, K.-D. Bouzakis, D. Sagris, and G. Mansour, "Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm," *Robotics and Computer-Integrated Manufacturing*, vol. 24, no. 1, pp. 50–59, 2008.

[5] Y. Zhang, M. C. Fontaine, V. Bhatt, S. Nikolaidis, and J. Li, "Multi-robot coordination and layout design for automated warehousing," in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, E. Elkind, Ed. International Joint Conferences on Artificial Intelligence Organization, 8 2023, pp. 5503–5511, main Track. [Online]. Available: https://doi.org/10.24963/ijcai.2023/611

[6] S. Mitsi, K.-D. Bouzakis, D. Sagris, and G. Mansour, "Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm," *Robotics and Computer-Integrated Manufacturing*, vol. 24, no. 1, pp. 50–59, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584506001025

[7] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[8] C. Liu and M. Tomizuka, "Real time trajectory optimization for nonlinear robotic systems: Relaxation and convexification," *Systems & Control Letters*, vol. 108, pp. 56–63, 2017.

[9] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.

[10] H.-C. Lin, C. Liu, and M. Tomizuka, "Fast robot motion planning with collision avoidance and temporal optimization," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018, pp. 29–35.

[11] C. Liu, C.-Y. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM Journal on Control and optimization*, vol. 56, no. 4, pp. 2712–2733, 2018.