Low center-of-gravity movement robotic arm with kinematic optimization algorithm for on-site construction

Ruiqi Jiang[1] and Xiao Li [1]

[1]Department of Civil Engineering, University of Hong Kong, HKSAR, China
richj233@connect.hku.hk, shell.x.li@hku.hk

Abstract –

Robotic arms have increasingly been applied to onsite building construction (e.g., bricklaying, welding, and 3D printing). However, engineers should make a great effort to customize different mobile platforms (e.g., quadruped, hexaploid, tracked robot) for robotic arms in the complex ground environment of the construction site. The cable-driven platform under the existing crane system has the potential to address this problem. Nevertheless, the robotic arm operation would cause the swing of the cable-driven platform. Therefore, this paper aims to propose a novel structure for the robotic arm, namely the center of gravity control margin (CoG-CM) robotic arm and its control algorithm to reduce the CoG movement in a cable-driven platform. Compared with the conventional robotic arm, the main contribution of the proposed one is that the arm consists of four parallel joints, which could provide the control capability of the system's CoG. This structure has infinite solutions for a determined target, and conventional control algorithms are unsuitable for this system. So, we formulate its CoG, and the robotic arm CoG pose control (ArmCoG-PC) is proposed to solve the kinematics of the CoG-CM robotic arm. Finally, the experiment on CoG-CM and typical 6-degree-of-freedom robotic arms validate our proposal.

Keywords –

Construction robots; Optimization algorithm; Robotic arm

1 Introduction

The contemporary construction site heavily depends on manual labor, exposing workers to substantial risks. In contrast, utilizing construction robots offers a solution for reducing human involvement in hazardous environments and presents the potential for productivity improvements in the construction industry [1]. Among the widely employed robotic technologies, the robotic arm is among the most prevalent [2]. Numerous researchers have made valuable outcomes of on-site construction applications, including additive manufacturing [3], automated installation [4-5], and robotic bricklaying [6].

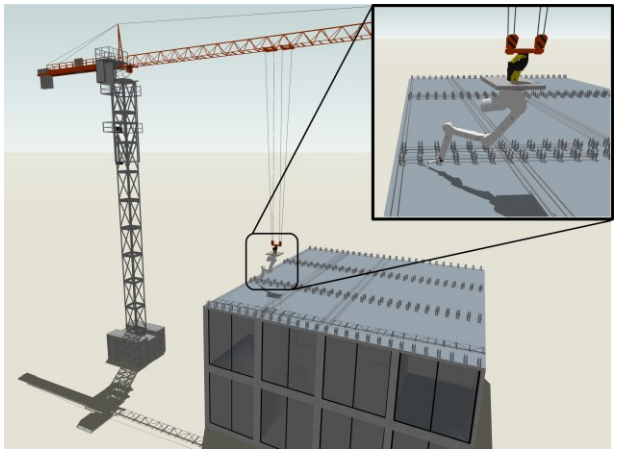


Figure 1. The robotic arm is mounted on a cable-driven platform under the existing tower crane.

While robotic arms show the capability in diverse construction tasks, their effective deployment relies on specially designed platforms for their application environments. Among the existing construction robots, wheeled platforms stand out as commonly utilized to support operational robotic arms [7]. Despite their adaptability to different environments, wheeled platforms face challenges when confronted with complex construction sites [8]. Therefore, these wheeled robots are mainly applied in the decoration stage with standard site environments, such as plastering [9] and fitting-out tasks [10]. To address these challenges, some researchers have innovatively designed rebar-tying robots sliding along tracks [11] and inspection robots moving on steel structures [12]. However, these designs require careful consideration of site conditions and constraints. As an alternative, researchers have explored the integration of aerial platform robotic arms into construction processes [13]. Nevertheless, drones' limited payload capacity restricts aerial robots' applicability in construction contexts.

Recently, the cable-driven platform (CDP) for robotic arms has been introduced into the construction domain [14]. The CDP is typically mounted at a tower crane above the working area, as shown in Figure 1. The CDP exhibits characteristics conducive to adapting to ground

conditions and supporting a promotive payload capacity [15]. Despite these advantages, the inherent elasticity of cables and the continual shifting of the center of gravity (CoG) during operations would cause a noticeable oscillation. This situation presents a challenge for robotic arms in achieving high-precision operations.

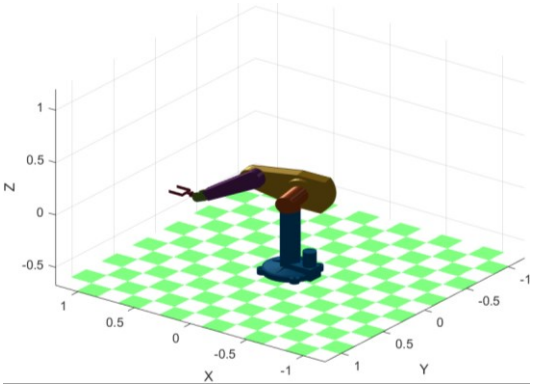


Figure 2. PUMA560 robotic arm.

To address the issue mentioned above, we propose a novel configuration of the robotic arm, namely the CoG control margin (CoG-CM) robotic arm, with a different joint sequence and amount from the conventional arm, as shown in Figure 2, along with its receding horizon control-based method, to minimize the movement of arm's CoG during operation. Specifically, the arm has four sequential parallel joints, providing a controllable margin for the CoG. However, the design causes higher computational complexity of joint pose. Traditional robotic arm pose control algorithms, e.g., [16] and [17], are not well-suited for solving this problem due to the highly coupled and nonlinear nature of the CoG formulation. In response, we propose the robotic arm CoG pose control (ArmCoG-PC) algorithm to tackle this issue. Within the ArmCoG-PC algorithm, the Particle Swarm Optimization (PSO) algorithm is introduced as a solver for optimal results, leveraging the principles of receding horizon control to achieve optimal outcomes within a sliding horizon. Additionally, we integrate constraints to effectively restrict the solution space to reduce the computational burden during the horizon solution stage.

The main contributions of this paper are as follows:

1. The CoG-CM robotic arm is proposed to reduce the CoG movement of the overall system, which is mounted on a cable-driven platform under the tower crane during the operation.

2. The control algorithm, namely robotic arm CoG pose control (ArmCoG-PC), is proposed. We also design constraints to limit the space of optimal results to reduce computational costs.

3. The comparison experiments are arranged, and the experimental results illustrate the effectiveness of the CoG-CM robotic arm and ArmCoG-PC algorithm.

2 Methodology

This section will first model the CoG Control Margin (CoG-CM) robotic arm using Denavit-Hartenberg (DH) representation parameters. Subsequently, the pose control algorithm, denoted as the robotic arm CoG pose control (ArmCoG-PC) algorithm, will be introduced. This algorithm is developed based on the CoG analysis and the state transition matrix.

2.1 Modeling CoG-CM robotic arm

To achieve a control margin for the CoG in the robotic arm, we propose a novel joint configuration called CoG-CM, which is different from wide-used robotic arms (e.g., PUMA560, as shown in Figure 2). In the CoG-CM robotic arm configuration, the first three joints are arranged sequentially perpendicular to each other, while the rotation directions of the third to sixth joints are parallel. Enhanced degrees of freedom (DoF) for the end effector are realized through the seventh and eighth joints.

The CoG-CM robotic arm could be denoted by DH representation [18]. The DH parameters are divided into standard and modified parameters and modeled based on links and joints as coordinate systems. The position of CoG is generally related to the positions of joints and the end of the links. Therefore, using the coordinate system at the end of the link, i.e., standard DH parameters, facilitates modeling the center of gravity for the CoG-CM robotic arm. Thus, the DH representation method in this paper represents the standard DH parameters. The CoG-CM robotic arm is shown in Figure 3, and its DH parameters are detailed in Table 1, where θ_i is the rotation angle of the ith joint, d_i is the distance along the z-axis from the ith joint to the $i+1$th joint, α_i is the twist about the x-axis between the ith and $(i+1)$, and a_i is the link length of the i th joint. To simplify the calculation, d_i are all set as 0.

Table 1 DH parameters of the CoG-CM robotic arm

Joint No.	θ	d	α	a
1	θ_1	0	0.5π	0
2	θ_2	0	0.5π	a_2
3	θ_3	0	0	a_3
4	θ_4	0	0	a_4
5	θ_5	0	0	a_5
6	θ_6	0	0.5π	a_6
7	θ_7	0	0.5π	0
8	θ_8	0	0.5π	0

The general form of the transformation matrix could be obtained from DH parameters as follows:

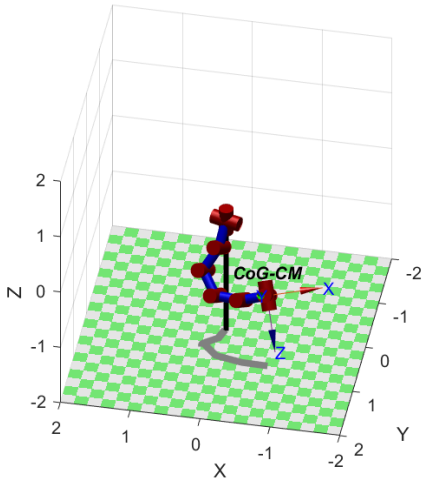


Figure 3. Center of gravity control margin (CoG-CM) robotic arm.

$$A_i^{i+1}(\theta_i)=\begin{bmatrix} R_i & O_i \\ 0 & 1 \end{bmatrix}$$

$$=\begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (1)$$

where matrix $A_i^{i-1}(\theta_i)$ describes the coordinate transformation between the ith and $(i+1)$th joints, R_i is the ith rotation matrix, O_i is the coordinate vector, $s(\cdot)$ and $c(\cdot)$ are sine and cosine functions, respectively.

Therefore, the transformation from the ith joint to i'th joint could be obtained by multiplying the transformation matrices as

$$A_i^{i'} = A_i^{i+1}\cdots A. \quad (2)$$

This paper mainly studies the rigidly connected robotic arms, so the CoG for the link between two adjacent joints remains relatively constant. Hence, the CoG for each link could be simplified. For ease of analysis, we consider each link as a uniformly distributed mass rod, and its CoG is located at the midpoint between two rotational joints. Specifically, for the link associated with the ith joint, with coordinates $O_i = [x_i\ y_i\ z_i]^T$, the CoG's coordinate O_{CoG_i} could be expressed as

$$O_{CoG_i} = \frac{1}{2}(O_i + O_{i+1}), \quad (3)$$

$$\begin{bmatrix} R_{i+1} & O_{i+1} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_i & O_i \\ 0 & 1 \end{bmatrix} A_i^{i+1}. \quad (4)$$

The position of the robotic arm's CoG could be denoted as

$$O_{CoG} = \frac{m_1 O_1 + m_2 O_2 + \cdots + m_8 O_8}{m_1 + m_2 + \cdots + m_8} \quad (5)$$

where m_i is the mass of the ith link.

2.2 ArmCoG-PC algorithm

The four parallel joints in the CoG Control Margin (CoG-CM) robotic arm will lead to an infinite kinematic solution for reaching one fixed target. Moreover, the Eq. (4) indicates the relationship between the CoG (O_{CoG_i}) of the ith link and the preceding joints (from θ_1 to θ_{i-1}), making the direct calculation of algebraic solutions challenging. Moreover, analyzing the Center of Gravity's (CoG) movement during operation requires a global perspective, making it challenging to constrain the CoG movement in a single-step planning. Consequently, the proposed control algorithm must be able to solve complex optimization problems and demonstrate a certain level of predictability. Receding horizon control, also known as model predictive control, involves establishing a prediction horizon and utilizing an optimizer to solve for the sequence of outputs within the horizon based on optimization objectives. Hence, we adopt the concept of receding horizon control to compute the kinematics of the CoG-CM robotic arm.

Firstly, the discrete state-space function of the robotic arm could be expressed as

$$\Theta(t+1) = A\Theta(t) + BU(t)$$

$$\Theta(t) = \begin{bmatrix}\theta_1(t) & \theta_2(t) & \cdots & \theta_8(t)\end{bmatrix}^T, \quad (6)$$

$$u_i(t) = \begin{cases} 1, \text{Clockwise} \\ 0, \text{No Rotation} \\ -1, \text{Counterclockwise} \end{cases}$$

where $\theta_i(t)$ represents the angle of the ith joint at time t, A is the identity matrix, B is a diagonal matrix, and $U(t) = [u_1(t) \ldots u_8(t)]^T$ is the rotation direction of joints at time t.

To maneuver the robotic arm end effector $P_{ed}(t) = [x_e(t)\ y_e(t)\ z_e(t)]$ towards the target point $P_{tr} = [x_p\ y_p\ z_p]$ while simultaneously minimizing the movement of CoG, the objective function could be designed as follows:

$$\min k_1 dis(P_{ed}(t), P_{tr}) + k_2 \sum_{t=0} dis[O_{CoG}(t), O_{CoG}(t-1)], \quad (7)$$

where $dis(.,.)$ represents the Euclidean distance between two points, and k_1 and k_2 are adjustable coefficients for the distance to the target and the movement of CoG, respectively. It is worth noting that

constraining the movement of the CoG throughout operation should be analyzed from the whole movement process. Only considering the output sequence obtained within a single step cannot guarantee that the result is optimal within a longer horizon.

Based on the above analysis, the pose control algorithm necessitates the capacity to consider every output in a certain horizon. Consequently, receding horizon control is introduced to minimize the movement of the robotic arm during operation. The receding horizon control could be outlined in the following steps:

Step 1 (Prediction): The system's future states are predicted using the current state and state-space function in a certain span (also called prediction horizon).

Step 2 (Optimization): Based on the predicted states, the control problem could be transferred to an optimization problem with constraints, and the aim is to find the optimal sequence of control inputs.

Step 3 (Implementation): Only the control input for the first step is implemented. This step-by-step approach ensures that the calculated control inputs are responsive to the evolving dynamics of the system.

The receding horizon control is widely used in certain systems owing to its robustness and low sensitivity. However, the challenges arise in our system due to multiple optimization objectives and high coupling, making the formulation of the optimization function a complex task. Therefore, it is necessary to modify the computation process for our model.

We propose a sequential computation approach to establish a predictive horizon for pose control and alleviate the complexity involved in formulating from the steps in the predictive horizon. Firstly, the pose for the last step is computed. Subsequently, the computation progresses backward to the first step within the constraints of each step's reachable solution. The ArmCoG-PC algorithm could be summarized as follows:

Step 1: At time t and the horizon length l, obtain the optimal results $\widehat{\Theta}$ of $(t+l)$th step based on Eq.(6) and objective function.

Remark 1: *It is noteworthy that, due to the constraints of window length l, the maximum available distribution range for the pose at the time* $(t+l)$ *is limited, i.e.,*

$$\Theta(t) - lB \le \hat{\Theta}(t+l) \le \Theta(t) + lB. \tag{8}$$

The optimal solution distribution space would be constrained. The objective function could be denoted as

$$\arg\min_{\hat{\Theta}(t+l)} f(t,l), \tag{9}$$

$$\begin{aligned} f(t,l) = {} & k_1 dis(P_{ed}(t+l), P_{tr}) \\ & + k_2 dis[O_{\mathrm{CoG}}(t), O_{\mathrm{CoG}}(t+l)]. \end{aligned} \tag{10}$$

Algorithm 1 The working procedure of PSO-RD

```
procedure PSO-RD(Input: Θ(t), l_i, and parameters of PSO-RD Output: Θ̂(t + l_i))
    Initialize parameters
    According to Eq. 8 or 11 to generate solution space X.
    while Not meet the set iteration do
        Update particle speed V, as equation .
        Constrain V with parameters.
        Update particle state S, as equation .
        Constrain S with solution space X.
        Update the best information of particle swarm.
    end while
    Output Θ̂(t + l_i) with the best information of particle swarm.
end procedure
```

Step 2: Based on the initial state $\Theta(t)$ and the obtained result $\widehat{\Theta}(t+l)$, the result of time $(t+l-1)$ is also limited and could be solved according to Eq. (9) and (10).

Remark 2: *As the movement of the robotic arm from state* $\Theta(t)$ *to state* $\widehat{\Theta}(t+l)$ *is a continuous process, the solution space for* $\widehat{\Theta}(t+l-1)$ *should be constrained by the two states, as illustrated in Figure 4, and it could be formulated as follows*.

$$\begin{cases} \Theta(t) - (l-1)B \le \Theta(t+l-1) \le \Theta(t) + (l-1)B \\ \hat{\Theta}(t+l) - B \le \Theta(t+l-1) \le \hat{\Theta}(t+l) + B. \end{cases} \tag{11}$$

Step 3: Similar to the ***Step 2***, compute the result of $\widehat{\Theta}(t+l-2)$ with $\Theta(t)$ and $\widehat{\Theta}(t+l-1)$ as constraints. Repeat this process iteratively until $\widehat{\Theta}(t+1)$ is obtained.

Step 4: Implement the output for time t with the Eq. (6) and $\widehat{\Theta}(t+1)$.

In practice, the primary controllers for robotic arms are digital and operate discretely. The control commands for the joints of the robotic arm are expressed as discrete sequences. Therefore, the distribution of each joint could be regarded as points in solution space. Heuristic or evolution algorithms could be adopted to solve the optimization problem of Eq.(9). This paper uses particle swarm optimization (PSO) as the optimizer. PSO algorithms are primarily designed for continuous solution spaces [19]. To adapt to the resolution of robotic arm's state, adjustments are necessary in terms of discretization and constraints. The PSO for robotic arm discrete problems (PSO-RD) is outlined in Algorithm 1, where l_i denotes the number of optimized steps. The speed and space update procedure could be expressed as Eq. (12) and (13), respectively, where v_i^n and s_i^n represent the velocity and state of particle i in the nth iteration, s_i^{best} and s^{gbest} signify the best states during the search process for the ith particle, and the swarm's best state, B' denotes the diagonal elements of B, and w_1, w_2, and w_3 are weight parameters.

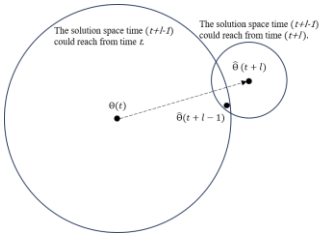


Figure 4. An example of the limitation of solution space.

$$v_i^n = round(w_1 v^{n-1} + w_2 rand(s_i^{best} - s_i^{n-1}) + w_3 rand(s^{gbest} - s_i^{n-1})), \quad (12)$$

$$s_i^n = s_i^{n-1} + B' v_i^n \quad (13)$$

The ArmCoG-PC algorithm could be expressed as Algorithm 2, where the symbol ⊘ represents each element of two vectors divided element-wise.

Algorithm 2 The working procedure of ArmCoG-CP optimization algorithm

procedure PSO-RD(**Input:** $\Theta(t), l$ **Output:** $\hat{\Theta}(t+1)$)
  Initialize parameters
  Compute $\Theta(t+l)$ with $\Theta(t)$ and Algorithm 1.
  for $(l-i) \geq 1$ **do**
    Compute $\Theta(t+l-i)$ with $\Theta(t+l-i+1)$, $\Theta(t)$ and Algorithm 1.
    $i = i - 1$.
  end for
  Set the output $U(t) = B' \oslash (\Theta(t+l) - \Theta(t))$.
end procedure

3 Experiment

This section conducts two sets of experimental comparisons: experiments involving the CoG-CM robotic arm model and a typical 6-DoF robotic arm model. For the CoG-CM robotic arm experiments, the comparison is made between the ArmCoG-PC algorithm and a direct joint rotation approach. Additionally, a comparison is made between the ArmCoG-PC algorithm and the commonly used inverse kinematics solution method on the typical 6-DoF robotic arm. Experiments only consider the mass of links to simplify the mass of robotic arms. The robotic toolbox in Matlab is used as the simulation environment [20].

3.1 Experiments on the CoG-CM robotic arm model

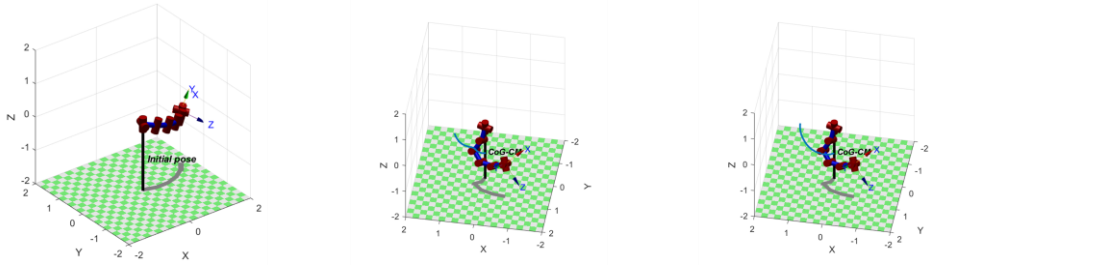
In this subsection, experiments based on the CoG-CM robotic arm model are analyzed, and the DH parameters of the CoG-CM robotic arm are shown in Table 2, where offset θ is the initial joint states expressed in radians.

The comparative process first employs the ArmCoG-PC algorithm to solve the kinematics of the CoG-CM robotic arm to reach the target. Subsequently, the joints are directly linearly rotated to achieve the final pose. The parameters of the PSO-RD comprise two types: particle parameters and coefficient parameters. The particle parameters significantly impact computational efficiency. Hence, we utilize an enumeration method, incrementing each parameter. Specifically, at each moment $t = 0$, ***Step 1*** of the ArmCoG-PC algorithm is used to assess the setting of particle parameters. For each iteration, 10-time-repeat experiments are conducted until the cost is minimized across all 10 experiments. The number of particles is set at 10, with 40 iterations. For enhancing the rounding precision in such a discrete system, the coefficients in PSO-RD, i.e., w_1, w_2 and w_3 are set as 0.8, 0.5, and 0.5, respectively. The coefficients of CoG movement and distance to the target k_1 and k_2 are all set as 1. The horizon length determines the algorithm's predictive capability and computational burden. We choose the value at which the motion trajectory appears as coherent as possible, i.e., 4. The initial states of the CoG-CM robotic arm $\Theta(0)$ = [-0.3, -0.3, -0.3, -0.3, -0.3, -0.3, -0.3, -0.3]. The target coordinates are [-0.7, 0.7, -0.7] and [-0.5, 0.5, -0.5]. The resolution of rotation is set as 0.02 rad.

The experimental results are presented in Figures 5, 6, and Table 3. Figures 5 and 6 depict the motion of the CoG-CM robotic arm optimized by the ArmCoG-PC algorithm and the direct linear rotation. The depicted blue lines represent the trajectory of the arm's CoG movement. The observations from Figures 5 and 6 illustrate that the poses devised by the ArmCoG-PC algorithm facilitate a more direct CoG movement than the linear direct rotation method, which follows a relatively longer curved path. Additionally, the CoG movement trajectory of the robotic arm, when planned with the ArmCoG-PC algorithm, exhibits a fluctuation as the endpoint of the robotic arm moves from one quadrant of the x-y plane to another, which means the proposed algorithm enables the arm to adjust its joints to maintain a low CoG movement distance. This observation also demonstrates the high CoG movement control margin of the CoG-CM robotic arm. Table 3 provides the total distance of the CoG movement for both methods. The results show the effectiveness of the ArmCoG-PC algorithm in significantly reducing the distance of CoG movement for the CoG-CM robotic arm.

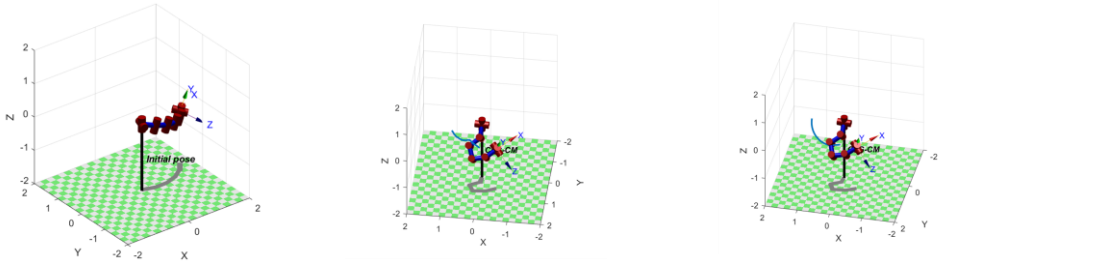
Table 2 DH parameters of the CoG-CM robotic arm in experiments

Joint No.	Offset θ	d	α	a
1	-0.3	0	0.5π	0
2	-0.3	0	0.5π	0.4
3	-0.3	0	0	0.4
4	-0.3	0	0	0.4
5	-0.3	0	0	0.4
6	-0.3	0	0.5π	0.4
7	-0.3	0	0.5π	0
8	-0.3	0	0.5π	0



(a) Initial pose (b)With ArmCoG-PC algorithm (c)With directly linear rotation method

Figure 5. Operation results of CoG-CM robotic arm to [-0.7, 0.7, -0.7].



(a) Initial pose (b)With ArmCoG-PC algorithm (c)With directly linear rotation method

Figure 6. Operation results of CoG-CM robotic arm to [-0.5, 0.5, -0.5].

Table 3 The movement distance of the CoG-CM robotic arm in experiments

Target	ArmCoG-PC	Linear Movement
[-0.7, 0.7, -0.7]	1.3250	1.5855
[-0.5, 0.5, -0.5]	1.2536	1.4843

3.2 Experiments on the typical 6-DoF robotic arm model

Table 4 The DH parameters of typical 6 DoF robotic arm in experiments

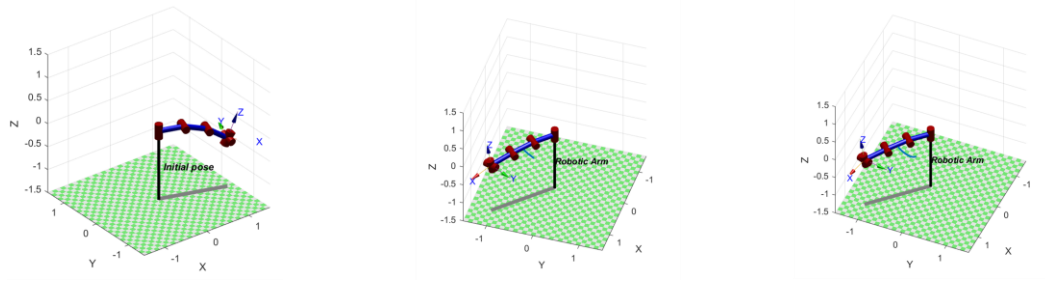
Joint No.	Offset θ	d	α	a
1	-0.3	0	0.5π	0.5
2	-0.3	0	0	0.5
3	-0.3	0	0.5π	0.5
4	-0.3	0	0.5π	0
5	-0.3	0	0.5π	0
6	-0.3	0	0	0

In this subsection, experiments based on the typical 6-DoF robotic arm are analyzed, and the DH parameters of the typical 6-DoF robotic arm are shown in Table 4. The setting of parameters is the same as in subsection 3.1. The initial states of the CoG-CM robotic arm $\Theta(0) =$ [-0.3, -0.3, -0.3, -0.3, -0.3, -0.3]. Due to the workspace limitation, the targets are set as [1, -1, -0.2] and [-0.2, 0.2, -0.4].

Figures 7 and 8 illustrate the movement of CoG to two different targets in two methods. As shown in Figure 7, the ArmCoG-PC optimization algorithm has an insignificant reduction of CoG movement due to the DoF constraints of the robotic arm. However, according to the data in Table 5, the ArmCoG-PC optimization algorithm still works in this situation, reducing almost 42% movement distance of the arm's CoG, compared with directly moving the joints. In Figure 8, for situations that require a larger rotation angle, the effectiveness of the ArmCoG-PC algorithm becomes more pronounced. With the direct linear rotation method, the trajectory of the arm's CoG movement corresponding to the overall motion follows a long curve, while the pose controlled by the ArmCoG-PC algorithm exhibits a straighter line. The case controlled by the ArmCoG-PC algorithm reduces the distance by 0.3979, approximately 37%. Compared to the cases controlled by the ArmCoG-PC algorithm in subsection 3.1, the CoG movement fluctuation in this subsection is much larger, further demonstrating the effectiveness of the CoG-CM robotic arm. Overall, these results show the notable effectiveness of the ArmCoG-PC algorithm in reducing CoG movement for the robotic arm under various operational conditions.

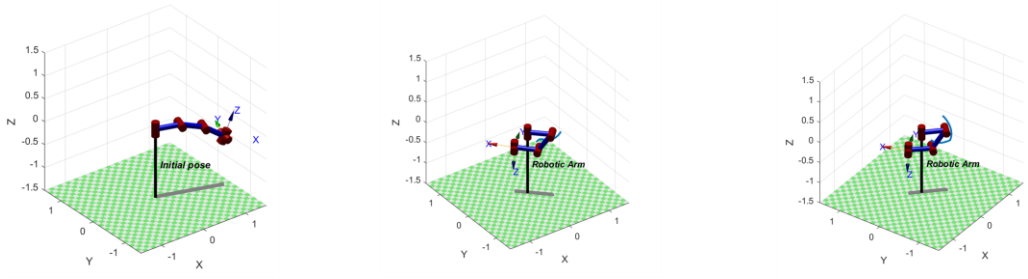
Table 5 The movement distance of the CoG-CM robotic arm in experiments

Target	ArmCoG-PC	Linear Movement
[1, -1, -0.2]	0.3400	0.5857
[-0.2, 0.2, -0.4]	0.6779	1.0758



(a) Initial pose (b)With ArmCoG-PC algorithm (c)With directly linear rotation method

Figure 7. Operation results of typical 6 DoF robotic arm to [1, -1, -0.2].



(a) Initial pose (b)With ArmCoG-PC algorithm (c)With directly linear rotation method

Figure 8. Operation results of typical 6 DoF robotic arm to [-0.2, 0.2, -0.4].

4 Conclusion

In this study, we addressed the challenge of oscillation induced by the Center of Gravity (CoG) movement in a robotic arm mounted on a cable-driven platform under the existing tower crane. Our study introduces the CoG Control Margin (CoG-CM) robotic arm configuration with a novel robotic arm CoG pose control (ArmCoG-PC) algorithm. The CoG-CM robotic arm, featuring four parallel joints, minimizes the CoG movement during operation. In the proposed configuration, the computational complexity and the requirement of predictive capability in motion control, as well as conventional algebraic kinematics methods, are unsuitable. Consequently, we propose the ArmCoG-PC algorithm to effectively solve the pose of the CoG-CM robotic arm. Experimental results, encompassing both the CoG-CM and a typical 6-DoF robotic arm, illustrate the robustness and efficiency of the ArmCoG-PC optimization algorithm.

While the effectiveness and advantages of the CoG-CM robotic arm and the ArmCoG-PC algorithm were demonstrated, certain aspects warrant further exploration in future studies. First, the experiments conducted thus far rely on simulations, and incorporating field testing would significantly bolster the applicability of our proposed approach to large-scale robotic arms in on-site construction. In addition, although longer link lengths in the CoG-CM robotic arm can improve the control margin, they may simultaneously reduce the arm's load capacity. An in-depth analysis of the relationship between link length and actual load capacities is needed to optimize the trade-off between control margins and overall robotic arm efficiency. Furthermore, when dealing with complex environments, the predictive horizon of the ArmCoG-PC algorithm should be expanded to ensure comprehensively low CoG movement. This expansion may cause a high computational cost, so future research should focus on methods for enlarging the predictive horizon without a proportional increase in computational costs.

For the on-site construction, the CoG-CM robotic arm and ArmCoG-PC algorithm could provide opportunities for large-scale (heavy) robotic arms, especially empowering those robotic arms mounted on the cable-driven platform with high-precise operation in on-site construction.

Acknowledgment

This research is supported by grants from the University of Hong Kong (HKU Project No.109000053 and No.006010313).

References

[1] Chen X, Chang-Richards A Y, Pelosi A, et al. Implementation of technologies in the construction industry: a systematic review[J]. Engineering,

Construction and Architectural Management, 2022, 29(8): 3181-3209. DOI: 10.1108/ecam-02-2021-0172.

[2] Gharbia M, Chang-Richards A, Lu Y, et al. Robotic technologies for on-site building construction: A systematic review[J]. Journal of Building Engineering, 2020, 32: 101584. DOI: 10.1016/j.jobe.2020.101584.

[3] Panda B, Lim J H, Mohamed N A N, et al. Automation of robotic concrete printing using feedback control system[C]. ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction. IAARC Publications, 2017, 34. DOI: 10.22260/isarc2017/0037.

[4] Lee S, Gil M, Lee K, et al. Design of a ceiling glass installation robot[C]. Proceedings of the 24th International Symposium on Automation and Robotics in Construction. 2007: 247-252. DOI: 10.22260/isarc2007/0044.

[5] Chu B, Jung K, Lim M T, et al. Robot-based construction automation: An application to steel beam assembly (Part I) [J]. Automation in construction, 2013, 32: 46-61. DOI: 10.1016/j.autcon.2012.12.016.

[6] Yu S N, Ryu B G, Lim S J, et al. Feasibility verification of bricklaying robot using manipulation trajectory and the laying pattern optimization[J]. Automation in Construction, 2009, 18(5): 644-655. DOI: 10.1016/j.autcon.2008.12.008.

[7] Ha Q P, Yen L, Balaguer C. Robotic autonomous systems for earthmoving in military applications[J]. Automation in Construction, 2019, 107: 102934. DOI: 10.1016/j.autcon.2019.102934.

[8] Turner C J, Oyekan J, Stergioulas L, et al. Utilizing industry 4.0 on the construction site: Challenges and opportunities[J]. IEEE Transactions on Industrial Informatics, 2020, 17(2): 746-756. DOI:

[9] Jenny S E, Pietrasik L L, Sounigo E, et al. Continuous Mobile Thin-Layer On-Site Printing[J]. Automation in Construction, 2023, 146: 104634. DOI: 10.1109/tii.2020.3002197.

[10] Nlink. Drilly: Mobile drilling robot. On-line: https://www.nlinkrobotics.com/projects/drilly-mobile-drilling-robot, Accessed: 11/12/2023.

[11] Momeni M, Relefors J, Khatry A, et al. Automated fabrication of reinforcement cages using a robotized production cell[J]. Automation in Construction, 2022, 133: 103990. DOI: 10.1016/j.autcon.2021.103990.

[12] La H M, Dinh T H, Pham N H, et al. Automated robotic monitoring and inspection of steel structures and bridges[J]. Robotica, 2019, 37(5): 947-967. DOI: 10.1017/s0263574717000601.

[13] Munoz-Morera J, Maza I, Fernandez-Aguera C J, et al. Assembly planning for the construction of structures with multiple UAS equipped with robotic arms[C]. 2015 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, 2015: 1049-1058. DOI: 10.1109/icuas.2015.7152396.

[14] Hu R, Pan W, Iturralde K, et al. Construction Automation and Robotics for Concrete Construction: Case Studies on Research, Development, and Innovations[C]. ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction. IAARC Publications, 2023, 40: 683-690. DOI: 10.22260/isarc2023/0095.

[15] Iturralde K, Feucht M, Illner D, et al. Cable-driven parallel robot for curtain wall module installation[J]. Automation in Construction, 2022, 138: 104235. DOI: 10.1016/j.autcon.2022.104235.

[16] Fundamentals of robotic mechanical systems: theory, methods, and algorithms[M]. New York, NY: Springer New York, 2003. DOI: 10.1007/978-0-387-34580-2.

[17] Gan J Q, Oyama E, Rosales E M, et al. A complete analytical solution to the inverse kinematics of the Pioneer 2 robotic arm[J]. Robotica, 2005, 23(1): 123-129. DOI: 10.1017/s0263574704000529.

[18] Niku S B. Introduction to robotics: analysis, control, applications[M]. John Wiley & Sons, 2020. DOI: 10.1108/ir.2003.04930cae.002.

[19] Kayhan A H, Ceylan H, Ayvaz M T, et al. PSOLVER: A new hybrid particle swarm optimization algorithm for solving continuous optimization problems[J]. Expert Systems with Applications, 2010, 37(10): 6798-6808. DOI: 10.1016/j.eswa.2010.03.046.

[20] Corke P I. A robotics toolbox for MATLAB[J]. IEEE Robotics & Automation Magazine, 1996, 3(1): 24-32. DOI: 10.1109/100.486658.