ASME Journal of Mechanisms and Robotics
Online journal at:
https://asmedigitalcollection.asme.org/mechanismsrobotics

Maximizing the Throwing Distance of Robotic Manipulators: An Optimization Approach

André Gallant[1]
Dynamium,
Département de génie mécanique,
Université de Moncton,
Moncton, NB E1A 3E9, Canada
e-mail: andre.gallant@umoncton.ca

Clément Gosselin
Fellow ASME
Laboratoire de robotique,
Département de génie mécanique,
Université Laval,
Québec, QC G1V 0A6, Canada
e-mail: gosselin@gmc.ulaval.ca

Manipulators are increasingly being called upon to perform a wide range of tasks. This paper explores the maximal distance throwing task for robotic manipulators and shows that this characteristic can be incorporated in the kinematic design process. Indeed, knowing the maximum distance that a manipulator can throw objects is useful in determining the viability of certain throwing tasks it might be called upon to execute. This paper studies three optimization problems: optimizing the release state to maximize the throwing distance, optimizing the kinematic trajectory subject to position, velocity, acceleration, and jerk constraints, and finally optimizing the kinematic design of manipulators to maximize the workspace as well as the throwing distance. Three manipulator architectures are used as case studies for these optimizations: a planar RR, a spatial RRR, and a wrist-partitioned 6R manipulator. [DOI: 10.1115/1.4064570]

Keywords: kinematics, dynamics, and control of mechanical systems, mechanism synthesis and analysis, motion and manipulation planning, robot motion planning and control

1 Introduction

Robotic manipulators are increasingly being called upon to perform a wide range of tasks that extend beyond the conventional tasks found in traditional applications. One such type of task is the throwing of objects. Particularly, the maximal distance throwing task, where an object is thrown the farthest possible, is studied in this work.

Potential applications of this study include industrial automation and search and rescue. In industries where rapid placement of objects is essential, manipulators with throwing capabilities can be used for sorting or relocating items quickly. For instance, in recycling plants, such robots could be employed to separate materials based on their type and weight, throwing them into designated bins. In search and rescue missions, robots capable of throwing lifelines, medical supplies, or communication devices over long distances could be invaluable, especially in terrains where human access is risky or limited.

The maximal distance throwing task is one that is characteristic of the manipulator's design parameters, such as its link lengths and its actuators' capabilities. As such, for a given manipulator, the maximal distance throwing optimization need only to be solved once. Of course, the purpose of most manipulators is not merely to throw objects but to perform a variety of tasks; throwing machines specifically designed for the throwing task offer the best throwing performance. However, in conjunction with other criteria, the maximal throwing distance capabilities of a manipulator can be used in the design process.

This work explores the distance throwing task, studies a method of finding the maximum distance throwing capabilities of robotic manipulators, and demonstrates that this characteristic can be used as an optimization goal in the design process. This is explored with case studies of three manipulator architectures, a planar RR manipulator, a spatial RRR manipulator, and a wrist-partitioned 6R manipulator. Proceeding in this manner, i.e., progressively increasing the complexity of the robot architecture, provides some useful insight into the increase of the throwing distance resulting from the introduction of additional joints, such as the base revolute joint and the wrist.

Throwing objects with robotic manipulators is not a new idea. Indeed, many researchers have studied the problem from several points of view. From a machine learning point of view, Aboaf et al. [1] studied the possibility of learning the task of throwing by adjusting the target when an error is detected instead of modifying the model of the manipulator. This is similar to a human's approach, if the throw is below the target, the next throw will be aimed higher. Many other researchers have also studied the problem from a learning perspective [2–4]. While great performance can be achieved with these methods, the learning process necessarily involves trial and error and when the task changes, a new learning process must be undertaken.

Another approach is to apply control theory to execute throwing motions. For example, Kato et al. [5] developed a controller that actively attempts to predict errors in the throwing distance and that releases the object earlier or later in the trajectory to

[1]Corresponding author.

A video summarizing the key findings of this paper can be found at youtu.be/lTFYUXKP1jY.

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the Journal of Mechanisms and Robotics. Manuscript received August 15, 2023; final manuscript received January 12, 2024; published online February 27, 2024. Assoc. Editor: Med Amine Laribi.

compensate. While the performance of such approaches is interesting, they are difficult to generalize to manipulators with more degrees-of-freedom (DOFs).

Many researchers have also studied the closely related tasks of juggling [6–13] and catching [14]. Others have studied how humans throw objects [15].

Several robots have been specifically engineered for the task of throwing. For example, there are mono-articular (1-DOF) robots that exhibit the ability to regulate multiple kinematic properties of the throw [16–19]. These properties include velocity, direction, and angular velocity, achieved by allowing the object to roll on the end-effector. In addition, underactuated double pendulum-type manipulators, equipped with nonlinear springs in one or both joints, have been designed with the express purpose of throwing objects [20–23]. Throwing tasks have also served as a platform to showcase the capabilities of planar 2-DOF manipulators equipped with variable viscoelastic joints [24].

Various motion generation techniques have been extensively investigated to optimize the task of precisely throwing an object. For instance, cubic splines in Cartesian space have been employed to dictate a throwing motion implemented on a 6-DOF manipulator [25]. Notably, Frank et al. [26] explored non-spherical object throwing models, providing a comprehensive analysis of ballistic effects such as drag, the Magnus effect, and stabilization of the cylinder's orientation through the application of a spin, much like throwing an American football. The optimization of throwing motion while maintaining balance using the zero-moment point on a complete humanoid robot model was studied by Kim [15]. Impressively, a humanoid robot was even used to deliver the opening pitch at a Major League Baseball game [27]. Zhang et al. [28] proposed that motion optimization procedures could be significantly accelerated by prioritizing easier-to-test constraints over more complex ones. This approach was demonstrated on a precision throwing task involving a 6-DOF industrial manipulator with various obstacles in its path. Moreover, the principle of dynamic sensitivity was employed to enhance the robustness of precision throws in the face of model uncertainties and zero position calibration errors [29,30].

Researchers have also considered alternative objective functions, such as energy conservation [31] and directional velocity [32].

From a design perspective, the kinematic and dynamic aspects of serial manipulators have been scrutinized with a specific task in mind [33]. A standout example is the design of a planar manipulator aimed at maximizing throwing distance [34]. Although this study was restricted to planar manipulators, it took into account not only the dimensions of each link but also the number of degrees-of-freedom. The study found that, under the given constraints, a 5-DOF planar manipulator provided the optimal throwing distance.

More recently, Erumalla et al. [35] conducted experiments with a 2-DOF planar manipulator to throw and catch a disk-shaped object. The end-effector of this manipulator was also disk-shaped, enabling the object to be thrown and caught without prehension. Bombile and Billard propose a control framework that enables a dual-arm robotic system to grab and toss objects onto a moving target [36] and Kasaei and Kasaei address the challenge of object throwing into a moving basket while avoiding obstacles by employing a deep reinforcement learning strategy [37].

To the authors' best knowledge, incorporating the distance throwing capabilities of a manipulator as a design consideration along with other objectives has not yet been explored in the literature. Furthermore, the design of spatial manipulators using distance throwing capabilities remains uncharted territory. Most studies examining the throwing task have focused exclusively on planar manipulators.

While a substantial body of work exists on the topic of robotic throwing, to the authors' best knowledge, these studies have not specifically addressed this application of interest. The incorporation of a manipulator's distance throwing capabilities as a design consideration, in conjunction with other objectives, remains largely unexplored in the literature. Moreover, the design of spatial manipulators with a focus on enhancing their distance throwing capabilities represents a relatively untapped area of research. It is noteworthy that the majority of studies investigating the throwing task have been predominantly focused on planar manipulators, leaving a gap in our understanding of more complex and versatile spatial manipulators.

This paper is structured as follows: Sec. 2 studies the throwing task and defines the optimization problems, Sec. 3 introduces the three manipulators used as case studies. In Secs. 4 and 5, an analysis of two maximal distance optimization problems is performed and a useful link between the two is found, Sec. 6 explores the distance throwing capabilities of a manipulator as a parameter in its design, and Sec. 7 concludes the paper.

2 Distance Throwing Task

2.1 Ballistic Models and Kinematics. To effectively assess the distance throwing capabilities of the manipulators examined in this study, we require a functional model of the object's ballistics. While this study does not consider the effects of drag and wind, we do employ two distinct models. These models range from a simplified version, which may prove beneficial in certain scenarios, to a more comprehensive one that could be required under different circumstances. Both models are depicted in Fig. 1. The first model (Fig. 1(*a*)) operates under the assumption that the object is projected from the origin of a given coordinate system. In contrast, the second model (Fig. 1(*b*)) accommodates an initial distance and elevation, both of which could potentially be negative.

The distance thrown in the first case can be computed as

$$D_s = \left|\frac{2v_x v_z}{g}\right|, \quad v_z \geqslant 0 \tag{1}$$

where v_x and v_z are the initial horizontal and vertical velocities, respectively, and g is the gravitational acceleration.

Since manipulators are built to move, the position of the end-effector at the time of release is generally not fixed. Therefore, the initial distance and the initial height are taken into consideration in the second model (Fig. 1(*b*)) in which case the distance thrown can be computed as

$$D_c = \left|x_0 + \frac{v_x}{g}\left(v_z + \sqrt{v_z^2 + 2gz_0}\right)\right| \tag{2}$$

where x_0 and z_0 are, respectively, the x and z coordinates of the initial throwing position.

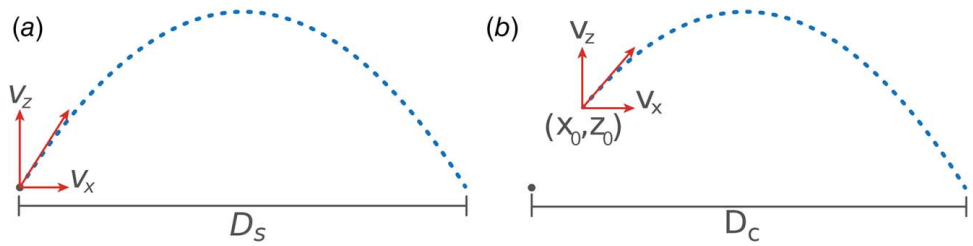

Fig. 1 Ballistic models: (*a*) simple ballistic model and (*b*) complete ballistic model

The initial distance and height (x_0, z_0) can be determined from the joint positions via direct kinematics. The velocity components in both models can similarly be computed employing the Jacobian matrix of the manipulator

$$\begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \mathrm{J}_v \\ \mathrm{J}_\omega \end{bmatrix} \dot{\mathbf{q}} \tag{3}$$

where $\mathbf{v}$ is the velocity of the reference point on the end-effector, $\boldsymbol{\omega}$ represents its angular velocity, $\mathrm{J_v}$ and J_ω are the translational and angular velocity components of the Jacobian matrix, respectively, and $\dot{\mathbf{q}}$ is the joint velocity vector.

In the context of the throwing task, we solely consider the velocity of the reference point on the end-effector, which is assumed to align with the object's center of mass. This results in

$$\mathbf{v} = \mathrm{J}_v \dot{\mathbf{q}} \tag{4}$$

For planar manipulators which operate in the vertical x–z plane, $\mathbf{v}$ directly yields the x and z components of the velocity:

$$\mathbf{v} = \begin{bmatrix} v_x \\ v_z \end{bmatrix} \tag{5}$$

In the case of spatial manipulators, the horizontal velocity splits into two components, $\mathbf{v}_x$ and $\mathbf{v}_y$. Similarly, the initial horizontal displacement possesses two components, x_0 and y_0. Moreover, the initial horizontal displacement vector and the horizontal velocity vector may not necessarily be collinear, as in a sidearm throw. Consequently, two components of the throwing distance must be computed (landing point)

$$\mathbf{d}_c = \begin{bmatrix} x_0 + \dfrac{v_x}{g}\left(v_z + \sqrt{v_z^2 + 2gz_0}\right) \\ y_0 + \dfrac{v_y}{g}\left(v_z + \sqrt{v_z^2 + 2gz_0}\right) \end{bmatrix} \tag{6}$$

Subsequently, the total throwing distance can be computed as

$$D_c = \sqrt{\mathbf{d}_c^{\mathrm{T}} \mathbf{d}_c} \tag{7}$$

While the direction of the throw can easily be determined from $\mathbf{d}_c$, it is not necessary to constrain the direction during the optimization. Most spatial manipulators feature a first joint with a vertical axis, allowing the direction of a throw to be conveniently controlled following any performed optimization.

2.2 Optimization Problems. This study investigates two distinct optimization problems. The first one focuses on optimizing the manipulator's state at the point of release. This problem imposes constraints solely on the state, specifically the joint position and velocity. The solution to this problem offers an upper limit on the throwing distance and can serve as a reference for subsequent optimizations. The second optimization task revolves around identifying the maximum throwing distance within the bounds of kinematic constraints. In this case, constraints including position, velocity, acceleration, and jerk must be adhered to throughout the entire trajectory.

2.2.1 Release State Optimization. The initial optimization problem examined in this study is also the most straightforward. The release state optimization problem (RSOP) is concerned with optimizing the manipulator's state at the precise moment the object is freed from its gripper. In essence, the trajectory preceding and succeeding the object's release is not considered in this segment of the research. The constraints for this optimization are limited to position and velocity.

Although this problem might not find direct application in most scenarios, the knowledge derived from its examination can yield valuable insights for ensuing optimizations. For example, the solution to this optimization offers a motion designer the maximum possible throwing distance of a given architecture, considering specific dimensions and joint limitations. Moreover, the solution to this problem can serve as an initial approximation for the more complex optimization problem discussed in the subsequent section.

Formally, the release state optimization problem can be defined as

$$\begin{aligned} &\max_{\mathbf{q}_R, \dot{\mathbf{q}}_R} D\left(\mathbf{q}_R, \dot{\mathbf{q}}_R\right) \\ &\text{s.t. } \mathbf{q}_{\min} \preceq \mathbf{q}_R \preceq \mathbf{q}_{\max} \\ &\quad \left|\dot{\mathbf{q}}_R\right| \preceq \dot{\mathbf{q}}_{\max} \end{aligned} \tag{8}$$

where $\mathbf{q}_R$ and $\dot{\mathbf{q}}_R$ denote the joint position and velocity vectors at the release moment, respectively. The objective function D can be either D_s or D_c from Eqs. (1), (2), or (7) and $\mathbf{q}_{\min}$, $\mathbf{q}_{\max}$, $\dot{\mathbf{q}}_{\min}$, and $\dot{\mathbf{q}}_{\max}$ represent the joint position and velocity constraints of the manipulator. The symbol $\preceq$ represents componentwise inequalities and the absolute value is also applied in a componentwise manner.

2.2.2 Trajectory Optimization With Kinematic Constraints. As previously discussed, the state of the manipulator at the release moment does not encapsulate all the critical considerations when researching object throwing with robotic manipulators. The presence of joint acceleration and jerk constraints may render the attainment of the optimal state unfeasible. Consequently, the trajectory both preceding and following the object's release deserves consideration.

In such a trajectory, the release timing during the trajectory emerges as an additional variable of interest. Consequently, the kinematic trajectory optimization problem (KTOP) can be articulated as

$$\begin{aligned} &\max_{\mathbf{q}(t), t_R} D(\mathbf{q}(t_R), \dot{\mathbf{q}}(t_R)) \\ &\text{s.t. } \mathbf{q}_{\min} \preceq \mathbf{q}(t) \preceq \mathbf{q}_{\max} \quad \forall\, t \in [0, T] \\ &\quad \left|\dot{\mathbf{q}}(t)\right| \preceq \dot{\mathbf{q}}_{\max} \quad \forall\, t \in [0, T] \\ &\quad \left|\ddot{\mathbf{q}}(t)\right| \preceq \ddot{\mathbf{q}}_{\max} \quad \forall\, t \in [0, T] \\ &\quad \left|\dddot{\mathbf{q}}(t)\right| \preceq \dddot{\mathbf{q}}_{\max} \quad \forall\, t \in [0, T] \\ &\quad t_R \leqslant T \\ &\quad \dot{\mathbf{q}}(0) = \mathbf{0} \\ &\quad \dot{\mathbf{q}}(T) = \mathbf{0} \end{aligned} \tag{9}$$

where t_R represents the release time and T is the total trajectory time.

For this study, we assume that the manipulator initiates and concludes the throwing motion at rest, as indicated in the final two constraints of Eq. (9). It is notable that the manipulator's state at the release time is the state used to compute the throwing distance. If the acceleration and jerk capabilities of the manipulator under study are sufficiently high, it might be feasible to identify a trajectory that achieves the upper bound provided by the solution to the RSOP, as described in Sec. 2.2.1.

3 Example Manipulators

To gauge the performance of the distinct optimization techniques, we applied each one to three progressively more complex manipulators, as depicted in Fig. 2.

The planar RR manipulator, illustrated in Fig. 2(*a*), is the first subject of our analysis, chosen for its relative simplicity. Thus, its examination may yield valuable insights.

Next, we study the spatial RRR architecture presented in Fig. 2(*b*). Given that the first revolute joint's axis of rotation runs parallel to the gravitational field, the direction of the throw can be readily determined and regulated. It is worth noting that the second and third links of this manipulator correspond to those of the planar RR manipulator (Fig. 2(*a*)). In this paper, we choose

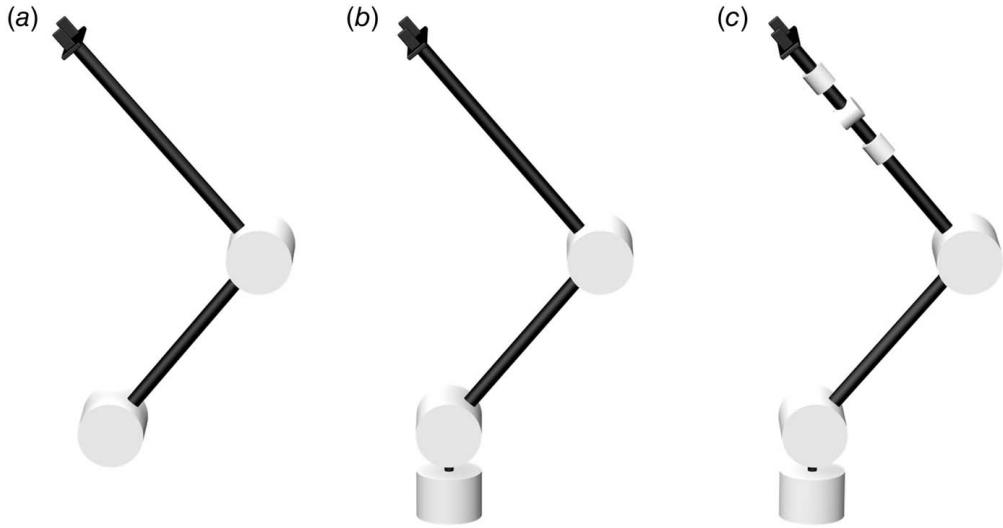

Fig. 2 The three manipulator architectures studied in this work: (*a*) RR planar, (*b*) RRR spatial, and (*c*) 6R spatial

link lengths such that immobilizing the first joint of the RRR manipulator effectively converts it into the RR manipulator.

The third manipulator under consideration in this study is a general wrist-partitioned 6R architecture, as shown in Fig. 2(*c*). The term 6R denotes a manipulator architecture consisting of six revolute joints linked in a series, a configuration common in many industrial manipulators. Here again, we select link lengths such that immobilizing the wrist joints of the 6R yields the kinematics of the RRR manipulator, and further immobilizing the first joint replicates the kinematics of the RR manipulator. By utilizing these architectures as case studies, we gain progressive insight into the impacts of the vertical first joint and the wrist.

4 Release State Optimization Problem

As elucidated in Sec. 2.2.1, the RSOP aims to optimize the state of the manipulator at the instant the object is released, with the objective of maximizing the throwing distance. This problem does not take into account the trajectory preceding or following the object's release.

This problem offers an opportunity to examine the variations in the throwing capacities of the three manipulators discussed in Sec. 3. Consequently, we can discern the contributions of the RRR manipulator's first joint and the 6R manipulator's wrist to the throwing distance. Moreover, the solution to this problem provides a maximum limit to the throwing distance, which could be instrumental in guiding future optimizations.

4.1 Nontrivial Solutions. At first glance, one might assume that solving the RSOP for the throwing task is straightforward. For instance, one might conjecture that the optimal release state for the RR manipulator should be a singular state where the manipulator is fully extended at a 45 deg angle with maximum joint velocities. This might indeed be the case if the joint velocity bounds are significantly high or if the simple D_s model from Eq. (1) is employed. However, in general, the impact of the end-effector's position at the moment of release cannot be overlooked. In other words, the robot's reach is not necessarily insignificant compared to the total throwing distance. To illustrate this, we can consider a planar RR manipulator with the following parameters:

$$L = \begin{bmatrix} 1.0\text{m} \\ 0.2\text{m} \end{bmatrix}, \quad \dot{\mathbf{q}}_{\max} = \begin{bmatrix} 2\text{rad/s} \\ 10\text{rad/s} \end{bmatrix} \tag{10}$$

where L is the array containing the two link lengths and $\dot{\mathbf{q}}_{\max}$ is the maximal joint velocity vector.

The solution to the RSOP for this manipulator is depicted in Fig. 3, where the two conventional 45 deg angle solutions are also displayed. It can be observed that the simple overhand solution does not achieve the same throwing distance as the optimal throw, and the simple underhand solution does not even manage to reach the floor from underneath.

While this robot may seem unrealistic, the distal link of this manipulator can be interpreted as a wrist. It is quite common for a manipulator's wrist to have shorter links and higher velocity bounds than the other links and joints. Indeed, this is certainly true of the human arm.

4.2 Objective Function and Constraints. In this study, we solve all optimization problems numerically, utilizing a sequential quadratic programming (SQP) algorithm as implemented by MATLAB's *fmincon* function. The equations used for the throwing distance are the complete models D_c in Eqs. (2) and (7).

As highlighted in Fig. 3, some throws do not reach the floor. This occurs when the square root $\sqrt{v_z^2 + 2gz_0}$ becomes imaginary. Consequently, the objective function becomes imaginary for a subset of the manipulator's state space. A natural solution to this issue is to consider the throwing distance as zero for the imaginary parts, as

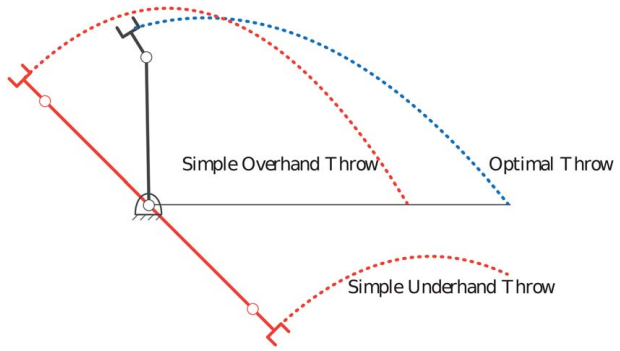

Fig. 3 The solution to the RSOP is not always trivial, even for a simple manipulator (note that the dimensions of this manipulator are not the same as those of the RR manipulator presented in Sec. 3)

expressed by

$$D_c = \begin{cases} 0 & \text{if } v_z^2 + 2gz_0 < 0 \\ x_0 + \frac{v_x}{g}\left(v_z + \sqrt{v_z^2 + 2gz_0}\right) & \text{otherwise} \end{cases} \tag{11}$$

However, this introduces a challenge for the numerical optimization because if the objective value for a given state (optimization guess) is zero and the objective value is also zero for all of its close neighbors, the computed gradient and Hessian matrix used in the SQP algorithm will also be zero. As a result, the optimization will fail due to the inability to compute a viable search direction.

To mitigate this issue, we can modify the objective function or add a constraint to avoid problematic states. The modified objective function used in this study is given by

$$D_c = \begin{cases} z_0 + \frac{v_z^2}{2g} & \text{if } v_z^2 + 2gz_0 < 0 \\ x_0 + \frac{v_x}{g}\left(v_z + \sqrt{v_z^2 + 2gz_0}\right) & \text{otherwise} \end{cases} \tag{12}$$

where $z_0 + (v_z^2/2g)$ corresponds to the distance between the peak of the ballistic trajectory and the ground. It is always negative when $v_z^2 + 2gz_0 < 0$ and provides a direct measure of how far the throw is from being valid. Thus, it nudges the optimization toward a valid throwing distance.

To demonstrate this modification of the objective function, consider the simple pendulum shown in Fig. 4(*a*). For simplification, we assume the joint velocity to be constant (maximum in the positive direction). This makes the objective function univariate and easily visualized. Figure 4(*b*) shows the objective value as a function of the joint angle $\theta \in [0, 360\,\text{deg}]$.

It can be observed that the original objective function (Eq. (11)) has large flat areas where the gradient is zero, i.e., where the object does not reach the floor from below. The modified objective function (Eq. (12)) guides the optimization toward a valid solution. This phenomenon is not exclusive to the simple pendulum; it applies to all the manipulators studied. Furthermore, when the object reaches the reference height, the modified and unmodified functions align. The two peaks of this function at $\theta = 65$ deg and $\theta = 153$ deg correspond to the maximum throwing distance in the positive and negative x direction, respectively. The bottom of the valley at $\theta = 101$ deg is due to the absolute value of Eq. (2) and corresponds to the object landing at the origin of the reference frame (the object is thrown in the negative x direction when $\theta > 101$ deg).

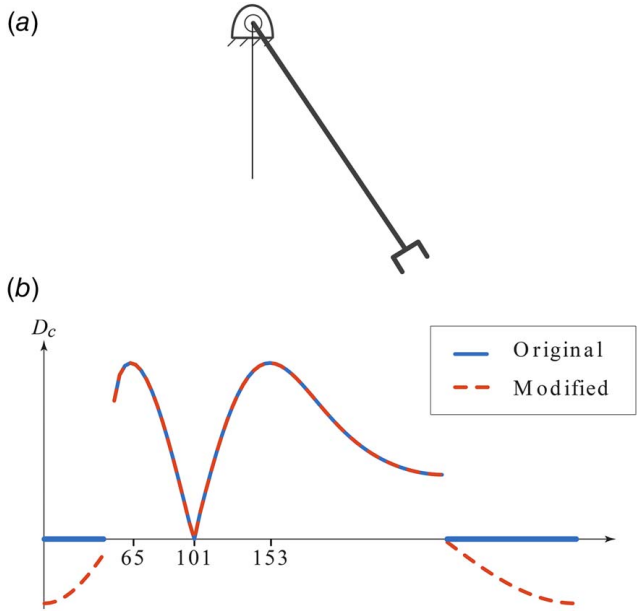

Fig. 4 Objective function of a simple pendulum throwing arm to show that there are flat parts and discontinuities: (*a*) pendulum and (*b*) objective functions

An alternative method of mitigating the issue with the flat objective function is to add a nonlinear constraint of the form

$$v_z^2 + 2gz_0 \geqslant 0 \tag{13}$$

In this study, a combination of the two objective functions (Eqs. (11) and (12)) with and without this nonlinear constraint are studied, as will be seen in Sec. 4.4.

4.3 Initial Guess. As can be discerned from Fig. 4(*b*), even for the uncomplicated pendulum in Fig. 4(*a*), the objective function is both non-convex and discontinuous. Consequently, the selection of the initial guess can greatly influence the convergence of the optimization process. There exist various strategies to refine the initial guess before initiating the optimization procedure.

First and foremost, one might conjecture that to maximize the throwing distance, the velocity of each joint should be maximized as well. For a predetermined manipulator pose, this hypothesis can be implemented via two different methods: the velocity direction for each joint can be chosen a priori (in this study, the positive direction is chosen) or the optimal combination of upper and lower bound velocities can be computed. The latter approach involves comprehensively evaluating the objective function with each combination of minimum/maximum joint velocities. This demands 2^n evaluations, where n is the number of degrees-of-freedom of the manipulator. For both methods, the initial pose of the manipulator is generated randomly, after which the joint velocity is determined. This supplies the initial guess utilized by the optimization procedure.

Another straightforward method for enhancing the quality of the initial guess is to evaluate a set of random guesses in advance, and to use the best performer as the initial guess for the optimization. The size of this set is a crucial parameter to consider. This study evaluates several set sizes, as will be discussed in Sec. 4.4.

4.4 Methodology and Results (RSOP). As discussed in Secs. 4.2 and 4.3, multiple ideas were formulated to study the behavior of the optimization. To assess these ideas, numerical experiments were executed using various parameters. Table 1 compiles the optimization procedures that were examined. Optimizations Opt1 through Opt4 apply the concepts from Sec. 4.2, while Opt5 through Opt11 incorporate the ideas from Sec. 4.3.

To assess the optimization procedures in Table 1, each procedure was run 1000 times, with the criteria detailed in Table 2 measured in each test. Moreover, these experiments were conducted on all three manipulators from Sec. 3.

These numerical experiments were conducted utilizing an Intel Core i5-4690 Haswell 3.5 GHz Quad-Core CPU in the MATLAB environment. The computation time for this optimization could be substantially reduced by utilizing a lower level, compiled programming language and quicker optimization algorithms. However, the

Table 1 Optimization parameters analyzed for the RSOP

Code	Objective	Initial guess
Opt1	Eq. (11)	Random
Opt2	Eq. (12)	Random
Opt3	Eqs. (11) and (13)	Random
Opt4	Eqs. (12) and (13)	Random
Opt5	Eq. (12)	Random pose, max velocity
Opt6	Eq. (12)	Random pose, best min/max velocity
Opt7	Eq. (12)	Best of random population of 10
Opt8	Eq. (12)	Best of random population of 50
Opt9	Eq. (12)	Best of random population of 100
Opt10	Eq. (12)	Best of random population of 300
Opt11	Eq. (12)	Best of random population of 1000

Table 2 Measures for the RSOP experiments

Code	Description
Best (m)	Best objective value after optimization
Average (m)	Average objective value after optimization
Average guess (m)	Average objective value before optimization
Success (%)	Rate of convergence within 5% of best
Time (ms)	Average computation time per optimization

Table 3 Manipulator link lengths

Manipulator	L (m)
RR	$[0.5;\ 0.6]$
RRR	$[0.1;\ 0.5;\ 0.6]$
6R	$[0.1;\ 0.5;\ 0.3;\ 0.1;\ 0.1;\ 0.1]$

study of numerical optimization algorithms is not the primary focus of this paper.

The link lengths of each manipulator are given in Table 3 and the maximum speed of each joint is fixed at 180 deg per second.

The results of the numerical experiments are compiled in Table 4. This table includes the results from all 11 optimization implementations from Table 1 and reports the measures outlined in Table 2.

From the data reported in Table 4, several observations can be drawn. The first is that the throwing distance capabilities are significantly augmented by the first joint of the RRR and 6R manipulators, as well as the wrist of the 6R manipulator. This enhancement occurs even though the first joint does not contribute to the vertical position or velocity, and despite the fact that the links in the wrist are relatively short and the total reach of all three manipulators is equal. The release state posture and throwing distance for each manipulator's optimal solution are illustrated in Fig. 5, highlighting the effect of the vertical joint and the wrist.

A second observation is that, given sufficient initial guesses and time, all optimization parameters can eventually converge to the best[2] solution.

It is also noteworthy that one of the most significant factors contributing to the convergence of the optimization is the absence of flat regions in the objective function, i.e., Eq. (12). In general, however, the optimal optimization appears to be the selection of an initial guess from a population of between 50 and 300 (Opt8–Opt10). Beyond that, the time required to evaluate the entire population increases significantly without yielding much improvement in the likelihood of obtaining the optimal solution. It is worth noting that the addition of the nonlinear constraint in Eq. (13) does provide an increase in the convergence rate, but at a significantly higher computational cost.

5 Kinematic Trajectory Optimization Problem

The solution to the preceding optimization problem (RSOP) offers a theoretical upper bound on the throwing distance capabilities of a manipulator, given the constraints on joint position and velocity. However, practical manipulators have additional kinematic constraints such as bounds on joint acceleration and jerk, which may render the RSOP solution unattainable.

Considering that a manipulator's trajectory is continuous, the dimensionality of the search space for the KTOP as defined in Eq. (9) from Sec. 2.2.2 is infinite. However, when constraints are imposed only on joint kinematics, the search space can be reduced to match that of the RSOP. For instance, when position, velocity, and acceleration bounds are imposed, the minimum displacement needed to accelerate to a specified joint velocity $\dot{q}_{R,i}$ at a constant joint acceleration $\ddot{q}_{\max,i}$ can be computed directly using

$$\Delta q_i = \frac{\dot{q}_{R,i}^2}{2\ddot{q}_{\max,i}} \tag{14}$$

This displacement is symmetric, i.e., the displacement required to accelerate from zero velocity is identical to the displacement needed to decelerate back to zero velocity. This symmetry applies even when the velocity is negative. Figure 6 illustrates this displacement.

In cases where a sudden change in acceleration from zero to its maximum and then back to zero could potentially harm the mechanical joints, and jerk constraints can be imposed. The minimum displacement Δq_i needed to reach a given velocity $\dot{q}_{R,i}$ under given maximum acceleration $\ddot{q}_{\max,i}$ and maximum jerk $\dddot{q}_{\max,i}$ constraints can still be explicitly calculated using

$$\Delta q_i = \frac{1}{2\ddot{q}_{\max,i}}\left(\dot{q}_{R,i} - \frac{\ddot{q}_{\max,i}^2}{\dddot{q}_{\max,i}}\right)^2 + \frac{\ddot{q}_{\max,i}}{2\dddot{q}_{\max,i}}\left(\dot{q}_{R,i} - \frac{\ddot{q}_{\max,i}^2}{2\dddot{q}_{\max,i}}\right) \tag{15}$$

which is obtained by integrating the position assuming that the acceleration follows a trapezoidal profile (constant jerk, constant acceleration, and constant negative jerk).

Therefore, the KTOP can be recast as

$$\begin{aligned} &\max_{\mathbf{q}_R, \dot{\mathbf{q}}_R} D(\mathbf{q}_R, \dot{\mathbf{q}}_R) \\ &\text{s.t. } |\dot{\mathbf{q}}_R| \preceq \dot{\mathbf{q}}_{\max} \\ &\quad \mathbf{q}_R - \Delta\mathbf{q} \succeq \mathbf{q}_{\min} \\ &\quad \mathbf{q}_R + \Delta\mathbf{q} \preceq \mathbf{q}_{\max} \end{aligned} \tag{16}$$

where $\Delta\mathbf{q}$ is the array containing Δq_i for each joint computed using either Eq. (14) or Eq. (15). This revised optimization problem is simpler to solve than the general KTOP in Sec. 2.2.2, and a complete kinematic trajectory can be readily computed from its solution.

5.1 KTOP Analysis. In the current work, Eq. (14) is utilized to calculate the components of $\Delta\mathbf{q}$. It should be pointed out, however, that for many manipulators the jerk bound is significantly large. Consequently, Δq_i computed using Eqs. (14) and (15) often yield closely similar results.

In order to grasp the influence of acceleration bounds on the maximum throwing distance capabilities of manipulators, an analysis of KTOP solutions is carried out with a range of acceleration bounds. Analogous to the procedure followed in the RSOP section above, for each manipulator, 1000 tests are executed, each with seven acceleration bounds ranging from 10 rad/s^2 to 0.5 rad/s^2. As the bounds decrease, it is expected that the maximum throwing distance will decrease correspondingly. For this test, the joint bounds imposed are summarized in Table 5, with the maximum joint speed set to 180 deg per second for each joint.

Moreover, the optimal solution to the RSOP found in the experiments reported in Sec. 4.4 has been employed as one of the initial guesses. Lastly, the optimization parameters chosen for this analysis are those from Opt9 in Sec. 4.4, i.e., Eq. (12) with the best of 100 random guesses chosen as the initial guess. The results are presented in Table 6, where the row "RSOP guess" reports the solution obtained when the RSOP solution is used as the initial guess.

Two key observations should be noted in these results. The first is that the success rate of this problem's optimization is notably inferior than that of the RSOP, particularly when the acceleration bounds are extremely strict. This suggests that the RSOP presents

[2]Since the optimization method employed in this work (SQP) is a gradient-based method, the global optimum cannot be guaranteed. However, considering the large number of optimizations performed in the analyses (11,000 for each manipulator), it is assumed that the best solution found is the global optimum.

Table 4 Results of the RSOP numerical experiments

	Opt1	Opt2	Opt3	Opt4	Opt5	Opt6	Opt7	Opt8	Opt9	Opt10	Opt11
RR manipulator											
Best (m)	3.22	3.22	3.22	3.22	3.22	3.22	3.22	3.22	3.22	3.22	3.22
Average (m)	1.74	3.00	2.95	3.20	2.07	2.23	3.13	3.22	3.22	3.22	3.22
Average guess (m)	0.34	0.17	0.35	0.20	0.77	1.13	1.17	1.65	1.86	2.18	2.47
Success (%)	52.6	90.7	91.4	99.5	64.1	69.3	96.1	100	100	100	100
Time (ms)	23.0	37.2	39.9	42.6	20.7	20.8	29.6	26.2	26.6	27.9	33.6
RRR manipulator											
Best (m)	4.23	4.23	4.23	4.23	4.23	4.23	4.23	4.23	4.23	4.23	4.23
Average (m)	2.51	3.93	4.12	4.14	2.92	2.94	4.14	4.21	4.23	4.22	4.22
Average guess (m)	0.43	0.26	0.41	0.22	1.20	1.64	1.33	1.87	2.12	2.49	2.85
Success (%)	57.1	90.2	95.8	96.6	67.6	69.2	96.9	99.3	100	100	100
Time (ms)	35.2	53.0	79.5	75.9	34.1	32.0	46.9	43.4	43.1	44.5	56.4
6R manipulator											
Best (m)	5.08	5.08	5.08	5.08	5.08	5.08	5.08	5.08	5.08	5.08	5.08
Average (m)	2.95	4.56	4.95	4.95	3.19	3.38	4.87	5.01	5.03	5.06	5.07
Average guess (m)	0.36	0.20	0.35	0.21	0.81	1.52	1.11	1.56	1.77	2.14	2.53
Success (%)	54.7	83.5	92.5	92.3	58.4	63.9	91.7	95.6	96.5	97.7	98.0
Time (ms)	98.1	150	265	267	97.7	95.4	136	137	137	151	201

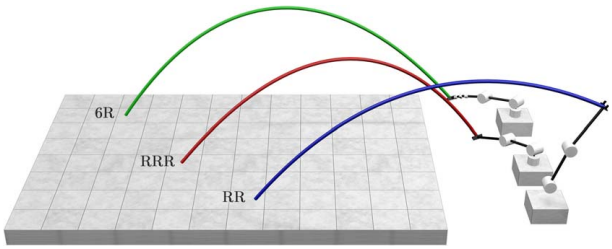

Fig. 5 Farthest throw solution to the RSOP for all three manipulators. The first joint and the wrist have a significant impact on the throwing distance capabilities of manipulators.

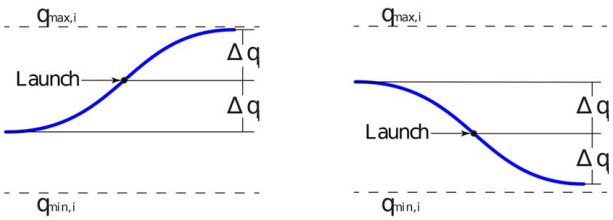

Fig. 6 The displacement required is the same for the acceleration phase and deceleration phase

a considerably simpler problem to solve. The second observation is that whenever the RSOP solution is utilized as an initial guess for the KTOP, the global optimum was consistently achieved. This is particularly compelling because it implies that the simpler RSOP can be employed to generate an initial guess that can then be used to solve the more complex KTOP.

It can also be observed that as the acceleration bounds tighten, the efficacy of the optimal throw correspondingly diminishes. Furthermore, it can be observed that more stringent acceleration bounds result in a significant decline in the success rate. This can be attributed to the substantial reduction in the subspace of solutions that satisfy all constraints.

Table 5 KTOP analysis joint bounds

Manipulator	$\mathbf{q}_{min/max}$ (deg)
RR	$\pm[360;\ 150]$
RRR	$\pm[360;\ 150;\ 150]$
6R	$\pm[360;\ 150;\ 150;\ 360;\ 150;\ 360]$

5.2 Additional Discussion on KTOP Results. To delve deeper into the results outlined in Table 6, we examine a specific case. This discussion focuses on a scenario involving a 6R manipulator, subject to a joint acceleration constraint of 1.00 rad/s^2. The optimization process, as detailed in Opt9 from Table 1, involved selecting 100 random points within the optimization design space. The optimal point from this set (characterized by the lowest) was then used as the starting point for the optimization process (best of 100). This method was replicated 1000 times to gather data on the success rate of convergence and the effectiveness of the random initial guesses. In this particular instance, the average "best of 100" demonstrated a throwing distance of 0.94 m prior to optimization, which improved to 1.81 m post-optimization. Notably, 49.7% of the 1000 trials successfully located the global optimum, which was identified as 2.88 m.

The analysis further explored the scenario where the solution to the RSOP is employed as the initial guess, instead of selecting the best of 100. In the case of the 6R manipulator, the initial

Table 6 KTOP analysis results using Opt9 from Table 1

Accel. (rad/s ^{2})	10.0	5.00	2.50	1.50	1.00	0.75	0.50
RR manipulator							
Best (m)	3.22	3.22	3.22	3.00	2.71	2.24	1.72
Average (m)	3.22	3.22	3.20	2.87	2.26	1.87	1.51
RSOP guess (m)	3.22	3.22	3.22	3.00	2.71	2.24	1.72
Success (%)	100	100	95.3	53.2	20.4	20.2	19.2
Time (ms)	28.4	29.5	30.0	33.5	33.4	37.9	44.2
RRR manipulator							
Best (m)	4.23	4.23	3.72	2.96	2.47	2.19	1.70
Average (m)	4.10	4.05	3.47	2.60	2.15	1.82	1.32
RSOP guess (m)	4.23	4.23	3.72	2.96	2.47	2.19	1.70
Success (%)	94.1	92.5	87.4	75.6	71.8	62.7	46.7
Time (ms)	56.7	47.4	52.7	63.1	73.3	77.3	78.0
6R manipulator							
Best (m)	5.08	5.08	4.53	3.54	2.88	2.50	1.89
Average (m)	4.75	4.68	3.94	2.78	1.81	1.09	0.48
RSOP guess (m)	5.08	5.08	4.53	3.54	2.88	2.50	1.89
Success (%)	87.0	86.0	78.4	65.0	49.7	31.7	16.0
Time (ms)	183	183	193	196	159	148	92.8

guess based on the RSOP solution was as follows:

$$\mathbf{q}_R = \begin{bmatrix} 0.06 \\ -0.22 \\ 1.47 \\ -2.19 \\ 0.29 \\ 0 \end{bmatrix}, \quad \dot{\mathbf{q}}_R = \begin{bmatrix} -3.14 \\ 3.14 \\ 3.14 \\ 3.14 \\ -3.14 \\ 0 \end{bmatrix} \tag{17}$$

Utilizing this solution as the initial guess in the optimization process, the subsequent results from the KTOP are as follows:

$$\mathbf{q}_R = \begin{bmatrix} -0.005 \\ -0.09 \\ 1.57 \\ -2.45 \\ 0.004 \\ 0 \end{bmatrix}, \quad \dot{\mathbf{q}}_R = \begin{bmatrix} -3.14 \\ 1.51 \\ 2.29 \\ -0.05 \\ 2.29 \\ 0 \end{bmatrix} \tag{18}$$

These results indicate that the solution derived from the RSOP differs from the optimal solution for the KTOP. However, the optimization process reliably identifies the global solution. Furthermore, the impact of the acceleration constraint is evident. The limited space available for the joints to reach full velocity before needing to decelerate, to avoid breaching the position constraint on the opposite side, is clearly observed.

6 Kinematic Design With Throwing Capacity

As previously articulated, the maximum throwing distance of a manipulator only needs to be determined once, as it is dependent solely on the design of the manipulator. However, it can also be deployed as a parameter in the kinematic design of the manipulator. Naturally, a multitude of factors are considered when architecting a new robotic manipulator. In this study, an example design of an RR manipulator and a 6R manipulator are optimized, taking into account both the reachable workspace and throwing capacity. The rationale for not incorporating the spatial RRR manipulator will be elucidated in the subsequent subsection. To constrain this problem, the total reach of the manipulator—that is, the summation of the link lengths—remains constant. Through this relatively simplified example, it is demonstrated that the throwing distance capabilities of a manipulator can indeed be incorporated in the design process.

6.1 RR/RRR Kinematic Design. The reachable workspace of a manipulator refers to the entire volume (or, in the case of planar manipulators, the area) that the end-effector is capable of accessing. For the planar RR manipulator, the reachable workspace essentially forms an annulus, bordered by two concentric circles. The area of the workspace can be calculated as

$$W_s = \pi\left[(L_1 + L_2)^2 - (L_1 - L_2)^2\right] \tag{19}$$

where W_s is the workspace area and L_1 and L_2 are the proximal and distal link lengths.

In the case of the RRR spatial manipulator, the workspace forms a region bounded by two concentric spheres. Note that the height of the first joint does not influence the size of the workspace. Therefore, an equation quite similar to the one used for the RR manipulator can be employed, except that it involves links 2 and 3. The volume of the workspace can be calculated as

$$W_s = \frac{4}{3}\pi\left[(L_2 + L_3)^3 - |L_2 - L_3|^3\right] \tag{20}$$

Indeed, this observation reveals that the co-optimization of the workspace and the throwing distance for the planar RR and spatial RRR manipulators presents striking similarities. Consequently, this paper will not engage in the optimization of the spatial RRR, as it is unlikely to provide any additional insights.

As indicated previously, the sum of the link lengths will be maintained as a constant parameter in order to constrain the problem, resulting in

$$W_s = \pi\left[L^2 - (2L_1 - L)^2\right] \tag{21}$$

where L is the sum of L_1 and L_2 and is constant. In a similar manner, the throwing distance can be computed as a function of L_1. Note that, for a given value of L_1, the value of each point in this function is the solution to one of the optimization problems discussed in this paper (RSOP or KTOP). From these two design criteria, namely the workspace and the throwing capacity, a weighted design objective function can be established as

$$f = AW_s + (1 - A)D_c \tag{22}$$

where $A \in [0, 1]$ is the weight of the workspace value with respect to the throwing distance.

Figure 7 shows the objective function as a function of L_1 for various values of A and with the design constraints summarized in Table 7. The computation of these curves is only done for illustration purposes since an optimization procedure would be used in a real design process. Indeed, the $\times$ on each curve indicates the optimal value as found by a single SQP optimization run with an initial guess of $L_1 = 0.5$ m. In any case, all these curves are convex and finding the optimum can be solved by a great number of algorithms.

As anticipated, when $A = 0$, only the throwing distance is considered. Consequently, the optimal value of L_1 is 0 which essentially creates a single pendulum with a maximum angular velocity of $\dot{q}_{\max,1} + \dot{q}_{\max,2}$. On the other hand, when $A = 1$, only the workspace of the manipulator is considered and the optimal design results in $L_1 = L_2$. Interestingly, the optimal design never has a value of L_1 exceeding 0.5. This is due to the fact that the angular velocity of the second link sums the velocities of both joints, thereby contributing more significantly to the maximum throwing distance. Clearly, this design process is rather simplistic as it considers only two objectives. If a larger number of design objectives were taken into account, the curves of the objective function would be significantly different.

6.2 6R Kinematic Design. As shown in Fig. 8, the decoupled 6R architecture studied in this paper only has three lengths that are relevant to the workspace and throwing distance (if L_1 is known), namely, L_2, L_{34}, and L_{56}. Indeed, the location of joint 4 on the L_{34} axis does not affect the end-effector position. The same can be said for joint 6 on the L_{56} axis.

The workspace of this manipulator architecture is either a sphere or a spherical shell whose outer radius is the sum of the link lengths (excluding L_1). The inner radius depends on the longest link and the

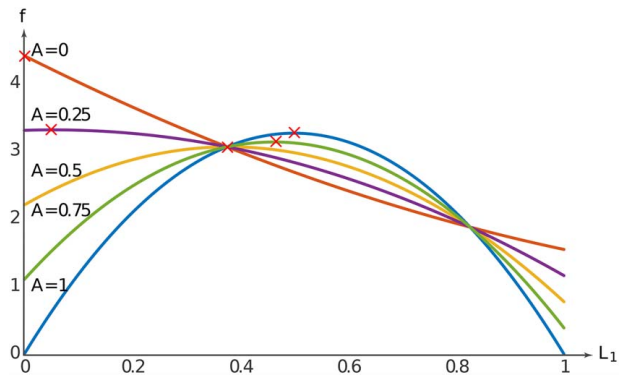

Fig. 7 Design curves with various values of *A* for the planar RR manipulator architecture

Table 7 RR dimensions and characteristics used in the design process case study

Parameter	Value
L	1 m
$\dot{\mathbf{q}}_{max}$	$[\pi; \pi]$ rad/s
$\ddot{\mathbf{q}}_{max}$	$[10; 10]$ rad/s ^{2}

sum of the lengths of the others. If none of the relevant lengths (L_2, L_{34}, L_{56}) is larger than the sum of the other two, there is no inner boundary and the workspace is a full sphere. This makes the workspace a constant function for a large part of the design space if the total length L is constant. Therefore, it is uninteresting as an optimization objective. The kinematic design of the 6R manipulator studied in this paper will, therefore, only optimize the throwing distance, with the following constraints:

(1) the total reach is constant at $L = 1$ m,
(2) none of the links (L_2, L_{34}, L_{56}) is longer than the sum of the others,
(3) no link is shorter than 0.1 m to keep the manipulator's structure reasonably intact.

Figure 9 shows a contour plot of the objective function as a function of L_{34} and L_{56}. Since the total reach is constant, $L_2 = L - L_{34} - L_{56}$. In this figure, the constraints are shown to illustrate which constraint is active at each of the bounds of the search space. Again, a × indicates the optimum as found by a single SQP run with an initial guess of $L_2 = L_{34} = L_{56} = L/3$. Again, this objective function is convex and easy to solve but this is only a simple example to show the distance throwing as a function of the kinematic parameters of the manipulator. The throwing distance could be added to any kinematic synthesis optimization.

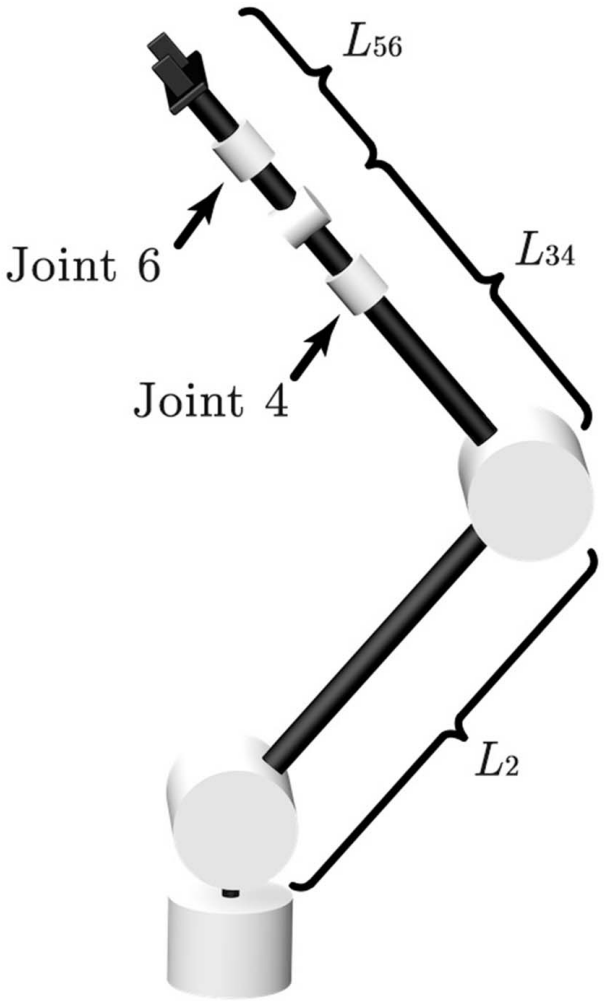

Fig. 8 Kinematic parameters of the studied decoupled 6R architecture

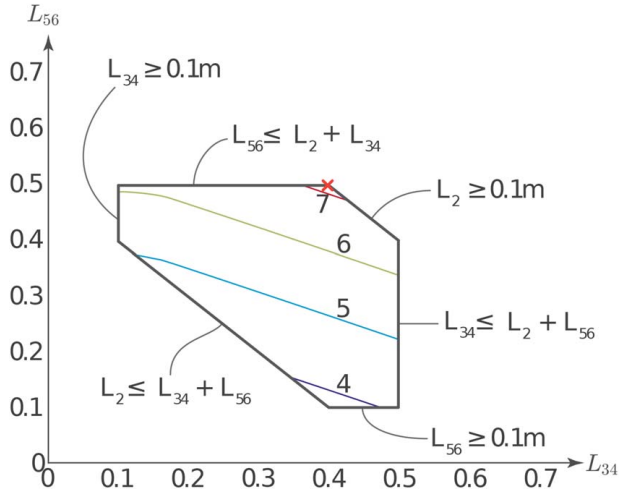

Fig. 9 Optimization function of the throwing distance for the kinematic design of the 6R manipulator

6.3 Discussion on Design Principles. In addressing the kinematic design of robotic manipulators for maximal distance throwing, it is essential to understand the fundamental principles that govern their motion and capabilities. Kinematics plays a pivotal role in robotic manipulator design. The fundamental objective in kinematic design is to determine the geometric arrangement of the manipulator's components to achieve the desired motion or performance. For throwing tasks, this involves optimizing the link lengths, and the sequence of movements to maximize the throwing distance.

The link lengths are crucial in defining the manipulator's reach and flexibility. In this optimization process, these parameters were varied to explore their impact on the throwing distance. It was observed that longer links increase reach but may increase the footprint of the manipulator. It was also observed that there is a fundamental compromise between throwing distance and workspace when a maximum footprint is imposed. The number of DoFs of a manipulator determines its ability to position and orient the end-effector. This study considers manipulators with varying DoFs, analyzing how this impacts the ability to optimize for maximal throwing distance. The trajectory of the end-effector is a critical aspect of kinematic design, especially for dynamic tasks like throwing. The approach in this paper involved plotting feasible trajectories that consider the limitations of joint velocities and accelerations, ensuring that the manipulator can achieve the desired throw without exceeding its physical capabilities. In future works, trajectory optimization methods involving dynamic constraints and smoothness are planned. The workspace of a manipulator defines the physical boundaries within which the robotic manipulator can operate. Since manipulators are called upon to perform various tasks, not only throwing objects, a compromise between workspace and other capabilities must be considered in diverse applications.

7 Conclusion

This study examines the determination of maximal throwing distance capabilities for robotic manipulators, using three example

designs: a planar RR manipulator, a spatial RRR manipulator, and a decoupled 6R manipulator. It further evaluates the potential for incorporating these throwing capabilities into the design strategy for manipulators.

First, it was shown that solving the maximal distance throwing problem is not as trivial as one would initially think, even for the simple planar RR manipulator. The common 45 deg angle launch is considerably sub-optimal. We also highlighted the significant contribution of factors such as the vertical position of the end-effector at launch, which cannot be overlooked in a more realistic ballistic model. To enhance the effectiveness of the optimization process, we proposed an approach that leverages the best initial guess from a set of random trials, offering a promising balance between computational time and convergence rate. Future work could further explore other global optimization strategies to refine this process.

The study delved into the role of the vertical first joint in RRR and 6R manipulators and the wrist of the 6R manipulator, which were found to significantly augment the throwing distance capabilities. These outcomes offer a fresh perspective on the design of robotic manipulators, given that the vertical first joint does not directly contribute to the vertical velocity of the end-effector, and the wrist links of the 6R manipulator are relatively short.

When considering the effect of joint bounds, it was shown that the velocity bounds must be high or the acceleration bounds must be low before a decrease in the distance throwing capabilities is observed. Solving the distance throwing problem with no kinematic constraints is much easier than solving it with all these constraints such as joint acceleration and jerk. Moreover, it was discovered that the solution to the easier, less constrained problem always resulted in finding the global solution to the more complex kinematically constrained problem, even when strict acceleration constraints were imposed.

Furthermore, our examination of the impact of joint bounds revealed that unless velocity bounds are high or acceleration bounds are low, the throwing distance capabilities are typically unaffected. It was also determined that the problem of maximal distance throwing with no kinematic constraints is more straightforward to solve than its counterpart with kinematic constraints such as joint acceleration and jerk. Interestingly, we found that the solution to the less constrained problem invariably led to the global solution of the more complex, kinematically constrained problem even when stringent acceleration constraints were in place.

This research provides insights into the complex relationship between the kinematic design of robotic manipulators and their throwing performance. It presents several promising strategies and considerations for the design and optimization of robots with specific task requirements, ultimately paving the way for more targeted and efficient robotic designs in the future.

Looking ahead, we plan to extend this research to encompass a broader range of design objectives and more complex manipulator models. We also intend to investigate the impact of dynamic constraints, such as joint torques and forces, to further enhance our understanding and optimization of robotic manipulator design.

References

[1] Aboaf, E. W., Atkeson, C. G., and Reinkensmeyer, D. J., 1988, "Task-Level Robot Learning," Proceedings of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, PA, Apr. 24–29, IEEE, pp. 1309–1310.

[2] Kolahdouz, M.-R., and Mahjoob, M. J., 2010, "A Reinforcement Learning Approach to Dynamic Object Manipulation in Noisy Environment," Int. J. Innov. Comput. Inform. Control, **6**(4), pp. 1615–1622.

[3] Kober, J., Öztop, E., and Peters, J., 2011, "Reinforcement Learning to Adjust Robot Movements to New Situations," international joint conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16–22, Vol. 22, p. 2650.

[4] Kim, S., and Doncieux, S., 2017, "Learning Highly Diverse Robot Throwing Movements Through Quality Diversity Search," Proceedings of the Genetic and Evolutionary Computation Conference Companion, Berlin, Germany, July 15–19, ACM, pp. 1177–1178.

[5] Kato, N., Matsuda, K., and Nakamura, T., 1996, "Adaptive Control for a Throwing Motion of a 2 Dof Robot," MIE Proceedings, 1996 4th International Workshop on Advanced Motion Control, 1996 (AMC'96), Vol. 1, Tsu-City, Mie-Pref., Japan, Mar. 18–21, IEEE, pp. 203–207.

[6] Buehler, M., and Koditschek, D. E., 1987, "Robotics in an Intermittent Dynamical Environment: A Prelude to Juggling," IEEE Conference on Decision and Control, Los Angeles, CA, Dec. 11.

[7] Aboaf, E. W., Drucker, S. M., and Atkeson, C. G., 1989, "Task-Level Robot Learning: Juggling a Tennis Ball More Accurately," 1989 IEEE International Conference on Robotics and Automation, Scottsdale, AZ, May 14–19, IEEE, pp. 1290–1295.

[8] Buehler, M., Koditschek, D. E., and Kindlmann, P., 1990, "Planning and Control of Robotic Juggling Tasks," Int. J. Rob. Res., **13**(2), pp. 101–118.

[9] Sakaguchi, T., Masutani, Y., and Miyazaki, F., 1991, "A Study on Juggling Tasks," Proceedings IROS'91. IEEE/RSJ International Workshop on Intelligent Robots and Systems' 91, Intelligence for Mechanical Systems, Osaka, Japan, Nov. 3–5, IEEE, pp. 1418–1423.

[10] Rizzi, A. A., and Koditschek, D. E., 1992, "Progress in Spatial Robot Juggling," Proceedings, 1992 IEEE International Conference on Robotics and Automation, 1992, Nice, France, May 12–14, IEEE, pp. 775–780.

[11] Buehler, M., Koditschek, D. E., and Kindlmann, P. J., 1994, "Planning and Control of Robotic Juggling and Catching Tasks," Int. J. Rob. Res., **13**(2), pp. 101–118.

[12] Sanfelice, R. G., Teel, A. R., and Sepulchre, R., 2007, "A Hybrid Systems Approach to Trajectory Tracking Control for Juggling Systems," 2007 46th IEEE Conference on Decision and Control, New Orleans, LA, Dec. 12–14, IEEE, pp. 5282–5287.

[13] Nguyen, H. N., and Olaru, S., 2013, "Hybrid Modelling and Constrained Control of Juggling Systems," Int. J. Syst. Sci., **44**(2), pp. 306–320.

[14] Nagendran, A., Crowther, W., and Richardson, R., 2011, "Dynamic Capture of Free-Moving Objects," Proc. Inst. Mech. Eng. Part I: J. Syst. Control Eng., **225**(8), pp. 1054–1067.

[15] Kim, J. H., 2011, "Optimization of Throwing Motion Planning for Whole-Body Humanoid Mechanism: Sidearm and Maximum Distance," Mech. Mach. Theory, **46**(4), pp. 438–453.

[16] Lynch, K. M., and Mason, M. T., 1996, "Dynamic Underactuated Nonprehensile Manipulation," Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems' 96, IROS 96, Osaka, Japan, Nov. 4–8, Vol. 2, IEEE, pp. 889–896.

[17] Miyashita, H., Yamawaki, T., and Yashima, M., 2009, "Control for Throwing Manipulation by One Joint Robot," IEEE International Conference on Robotics and Automation, 2009 (ICRA'09), Kobe, Japan, May 12–17, IEEE, pp. 1273–1278.

[18] Mori, W., Ueda, J., and Ogasawara, T., 2009, "1-dof Dynamic Pitching Robot That Independently Controls Velocity, Angular Velocity, and Direction of a Ball: Contact Models and Motion Planning," IEEE International Conference on Robotics and Automation, 2009 (ICRA'09), Kobe, Japan, May 12–17, IEEE, pp. 1655–1661.

[19] Mori, W., Ueda, J., and Ogasawara, T., 2010, "A 1-dof Dynamic Pitching Robot That Independently Controls Velocity, Angular Velocity and Direction of a Ball," Adv. Rob., **24**(5–6), pp. 921–942.

[20] Mettin, U., Shiriaev, A. S., Freidovich, L. B., and Sampei, M., 2010, "Optimal Ball Pitching With an Underactuated Model of a Human Arm," 2010 IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK, May 3–7, IEEE, pp. 5009–5014.

[21] Shoji, T., Nakaura, S., and Sampei, M., 2010, "Throwing Motion Control of the Springed Pendubot Via Unstable Zero Dynamics," 2010 IEEE International Conference on Control Applications (CCA), IEEE, pp. 1602–1607.

[22] Mettin, U., and Shiriaev, A. S., 2011, "Ball-Pitching Challenge With an Underactuated Two-Link Robot Arm," IFAC Proc. Vol., **44**(1), pp. 11399–11404.

[23] Yedeg, E. L., and Wadbro, E., 2013, "State Constrained Optimal Control of a Ball Pitching Robot," Mech. Mach. Theory, **69**(1), pp. 337–349.

[24] Tomori, H., Majima, T., Ishihara, H., and Nakamura, T., 2017, "Throwing Motion With Instantaneous Force Using a Variable Viscoelastic Joint Manipulator," J. Intell. Mater. Syst. Struct., **28**(8), pp. 999–1009.

[25] Lombai, F., and Szederkényi, G., 2009, "Throwing Motion Generation Using Nonlinear Optimization on a 6-Degree-of-Freedom Robot Manipulator," IEEE International Conference on Mechatronics, 2009 (ICM 2009), Malaga, Spain, Apr. 14–17, IEEE, pp. 1–6.

[26] Frank, H., Mittnacht, A., and Scheiermann, J., 2009, "Throwing of Cylinder-Shaped Objects," IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2009 (AIM 2009), Singapore, July 14–17, IEEE, pp. 59–64.

[27] Lofaro, D. M., Sun, C., and Oh, P., 2012, "Humanoid Pitching at a Major League Baseball Game: Challenges, Approach, Implementation and Lessons Learned," 2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids), Osaka, Japan, Nov. 29–Dec. 1, IEEE, pp. 423–428.

[28] Zhang, Y., Luo, J., and Hauser, K., 2012, "Sampling-Based Motion Planning With Dynamic Intermediate State Objectives: Application to Throwing," 2012 IEEE International Conference on Robotics and Automation (ICRA), Saint Paul, MN, May 14–18, IEEE, pp. 2551–2556.

[29] Okada, M., Pekarovskiy, A., and Buss, M., 2015, "Robust Trajectory Design for Object Throwing Based on Sensitivity for Model Uncertainties," 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, May 26–30, IEEE, pp. 3089–3094.

[30] Okada, M., Oniwa, S., and Hijikata, W., 2018, "Robust Throwing Design Based on Dynamic Sensitivity Analysis," Mech. Eng. J., **5**(1), pp. 17–00442.
[31] Sato, A., Sato, O., Takahashi, N., and Kono, M., 2007, "Trajectory for Saving Energy of a Direct-Drive Manipulator in Throwing Motion," Artif. Life Rob., **11**(1), pp. 61–66.
[32] Senoo, T., Namiki, A., and Ishikawa, M., 2008, "High-Speed Throwing Motion Based on Kinetic Chain Approach," IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, Sept. 22–26, IEEE, pp. 3206–3211.
[33] Patel, S., and Sobh, T., 2015, "Task Based Synthesis of Serial Manipulators," J. Adv. Res., **6**(3), pp. 479–492.
[34] Miyazaki, T., and Sanada, K., 2017, "Experimental Validation of an Optimum Design Method for a Ball Throwing Robot Considering Degrees of Freedom, Link Parameters, and Motion Pattern," JSME Mech. Eng. J., **4**(5), pp. 17–00147.
[35] Erumalla, S. R., Pasupuleti, S., and Ryu, J.-C., 2018, "Throwing, Catching, and Balancing of a Disk With a Disk-Shaped End Effector on a Two-Link Manipulator," ASME J. Mech. Rob., **10**(5), p. 054501.
[36] Bombile, M., and Billard, A., 2023, "Bimanual Dynamic Grabbing and Tossing of Objects Onto a Moving Target," Rob. Auton. Syst., **167**, p. 104481.
[37] Kasaei, H., and Kasaei, M., 2023, "Throwing Objects Into a Moving Basket While Avoiding Obstacles," IEEE International Conference on Robotics and Automation (ICRA), London, UK, May 29–June 2, IEEE, pp. 3051–3057.