IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Planning With Purpose: Task-Specific Trajectory Optimization

Yinan Pei and Yuri Ivanov

***Abstract*—In this letter, we propose an approach to trajectory planning based on the purpose of the task. For a redundant manipulator, many end effector poses in the task space can be achieved with multiple joint configurations. In planning the motion, we are free to choose the configuration that is optimal for the particular task requirement. Many previous motion planning approaches have been proposed for the sole purpose of maximizing manipulability, or minimizing effort. However, there is a lack of formulation that is flexible enough to allow the designer to purposefully define the motion and force priority of the planned trajectory. Our approach exploits both velocity and force manipulability, depending on the purpose of the task. In this formulation, the purpose of the task is defined by the motion preference ("fast" or "strong"), which can be characterized by a direction of the desired motion, or force. These two directions can be used to evaluate the compatibility of a chosen configuration with the given task. We first demonstrate the possibility of generating two distinct motion plans by the kinematic alignment of desired velocity and force directions with the manipulator's velocity and force manipulability ellipses. Next, this configuration selection strategy is incorporated into a task-specific trajectory optimization formulation to generate dynamically feasible trajectories. Two distinct motions (force-oriented lifting motion and velocity-oriented ballistic motion) are planned. We also propose a blending method to generate a single motion plan that considers both force and velocity, each to a specified degree. Together the three motions (force, velocity, and blended) are successfully planned and executed on a three-link serial robotic manipulator. The letter concludes with discussion and future directions.**

***Index Terms*—Manipulability, motion planning, pose optimization, redundant manipulator, trajectory optimization.**

I. Introduction

A. Why Planning With Purpose?

STATE-OF-THE-ART robot motions are not efficient. Robot programmers rarely directly consider the tasks that robots will be executing, and develop fairly generalized motion paths, focusing only on reachability and rated torques and velocities. In this sort of planning it is typically assumed that a robot can be strong and fast enough to perform any task it encounters in an application, which results in oversized robotic systems that are more expensive and dangerous around people. In contrast, humans often take on and succeed at physical tasks that far exceed their typical performance through purposeful motion planning. To build smaller, cheaper, and safer robots, we would like to look at the sources of these types of extended human abilities, and understand the basis for the extreme performance. In this letter, with a lot of hope for the future, we present some results of our first experiments in that direction.

Human body is a highly redundant system, which enables us to choose the most suitable body or arm pose to perform an intended task. Picking the "optimal" pose is based on the purpose of the task. For example, in weightlifting, to achieve the maximum lifting capacity, the athlete would keep the barbell close to the body, align the joints into a series of singular configurations, and exert impulsive forces to move the weight through these singularities to finally reach the fully extended upright stance. In each of these configurations, most of the weight is supported by the body structure, rather than the muscles. Conversely, to achieve the maximal shot velocity, a badminton player will first flex their arm around the shoulder and then quickly extend it in a swing. In these examples, humans deploy a motion trajectory through a sequence of joint poses that are most advantageous for minimizing muscle force (joint torque) in a weightlifting task, or for maximizing linear velocity of the hand (endpoint velocity) in a ballistic task. We would like to explore the effect of trajectory "planning with purpose" to enable a robotic manipulator to achieve velocity- or torque-efficient motion in the extremes of the range of its performance.

B. Motion Planning for Redundant Manipulators

To allow manipulation redundancy, industrial manipulators often have 7 degrees of freedom (DOFs), while the pose of a rigid object in 3D Cartesian space is uniquely defined by only 6 variables (three positions and three orientations). Ignoring for the time the end effector (EE) orientation, in 2D workspace, a three-link serial manipulator will be redundant and provide a simplified experimentation platform that we use in this discussion. When a redundant manipulator is away from its singularities, for a given target in the workspace, there will be infinitely many solutions (or configurations)[1] in the joint space. Even though all these configurations reach the same workspace target, depending on the nature of the task (load-bearing or moving fast), some joint poses might be more advantageous than others. Therefore, in this study, we would like to not only plan a feasible workspace trajectory from the start to the target, but also consider whether the pose is suitable for the intended task along the trajectory.

Manuscript received 30 November 2023; accepted 29 February 2024. Date of publication 22 March 2024; date of current version 4 April 2024. This letter was recommended for publication by Associate Editor P. Di Lillo and Editor J. P. Desai upon evaluation of the reviewers' comments. *(Corresponding author: Yuri Ivanov.)*

The authors are with Amazon Fulfillment Technologies and Robotics, North Reading, MA 01864 USA (e-mail: yivanov@gmail.com).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3381013, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3381013

[1] In this work, "pose" and "configuration" are used interchangeably, and apply to both joint and task spaces, when is clear from context.

For a target defined in the workspace, to find the corresponding joint poses, one can solve an Inverse Kinematics (IK) problem. Common techniques include iterative method [1], Jacobian-based method (e.g., resolved-rate control [2], null-space projection [3]), and others. The challenges of the IK approach are the intensive computational cost and the difficulty to incorporate manipulator dynamics, especially for high-DOF manipulators. As opposed to solving the IK problem, resolving the joint poses for desired workspace targets could also be formulated as a forward-planning problem via sampling-based methods, such as [4], and nonlinear optimization [5]. Since our target motions are highly dynamic and we need to evaluate the pose compatibility along the task-specific trajectory, the trajectory optimization approach that allows a straight-forward formulation to define dynamics constraint and cost on joint pose selection becomes a natural choice.

C. Trajectory Optimization

Trajectory optimization formulates the motion planning as a problem of optimal control, and aims to find feasible trajectories to minimize the user-defined cost function, while respecting the equality (e.g., dynamics constraint) and inequality (e.g., actuator torque and velocity limits) constraints [6]. This approach has been widely used in the planning of highly dynamic systems such as legged robots [7], etc. Trajectory optimization can find a feasible trajectory, but it is not necessarily stable. The obtained trajectory tracking can be susceptible to model inaccuracy, external disturbance, etc., so it is often necessary to stabilize the trajectory with a feedback controller [8]. Furthermore, in order to augment this formulation with pose awareness in its cost function, we need to find the appropriate metric to evaluate different joint poses for a given task. For this purpose we selected the concept of *manipulability*.

D. Manipulability

Manipulability is a well-established metric to evaluate the robot EE's ability to operate in any point of its joint space [9]. For any joint configuration, manipulability can be visualized intuitively in the robot workspace via a velocity ellipsoid (Fig. 1). The ellipsoid represents the mapping from unit norm joint velocity to EE's linear/angular velocity through manipulator's Jacobian (J) at a given set of joint angles. As the robot approaches a kinematic singularity, its Jacobian loses rank, resulting in loss of ability to translate or rotate in one or more directions. Therefore, driving a robotic manipulator into a singular configuration has been considered unfavorable and should be avoided to maximize its manipulability measure [10]. On the other hand, humans don't shy away from body singularities, and instead constantly use them in their favor. The benefit of the singularity is captured by a complementary metric, the force ellipsoid, which is the reciprocal of the velocity ellipsoid and represents the mapping between the unit norm joint torque and force/torque at EE in the workspace. In singularity, the radius of the velocity ellipsoid reduces to zero in the direction where the manipulator can no longer move. In contrast, that of the force ellipsoid stretches to infinity, indicating that in that exact joint configuration no joint torque is required for sustaining infinite workspace force/torque along that direction since all the force is absorbed entirely by the robot's mechanical structure.

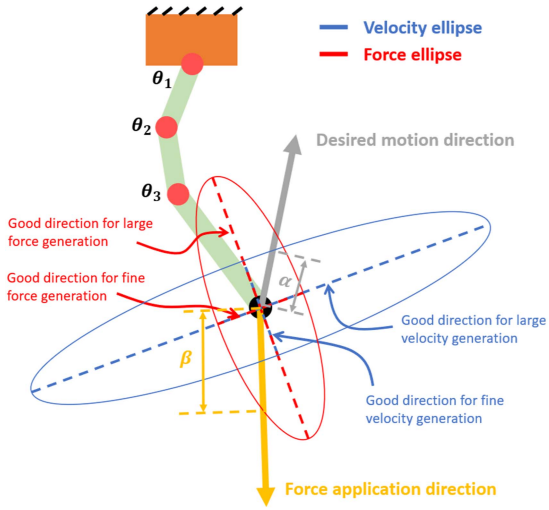

Fig. 1. Illustration of a manipulator's velocity and force ellipses at a certain pose with mock motion and force directions.

E. Related Work

Chiu [11] recognized that velocity and force ellipsoids provided insights on optimal directions to perform certain tasks. This idea was especially applicable to redundant manipulators since the redundancy enabled the possibility of choosing the suitable poses depending on the task. These insights led to research in task-specific kinematic planning. Lee [12] abstracted the task-specific motion and force reference trajectories as a series of desired velocity ellipsoids for the manipulator to track. Jun et al. [13] introduced task-specific objective function to resolve static optimal manipulator poses for maximal translational manipulability. Lee et al. [14] proposed an optimization method to select the task-specific reaching pose of a humanoid robot that achieves high arm manipulability or low energy consumption. Jaquier et al. [15] proposed a geometry-aware encoding in terms of symmetric positive definite manifolds to define a set of desired velocity ellipsoids during manipulation demonstration and used this metric as cost function to resolve robot configuration to track the reference ellipsoids. However, these studies as well as many others [10] mainly focused on maximizing manipulability to facilitate motion tasks and did not consider the reduced force capability (maximal at singularity), except for [12]. This aspect received less attention (e.g., [16]) since it often involves singularities. Furthermore, these planning approaches did not explicitly consider system dynamics and only provided kinetostatic motion plans, which might not apply to highly dynamic motions.

F. Study Overview

In this study, we propose an approach that incorporates the metric of velocity and force ellipsoids into an offline trajectory optimization formulation to generate task-specific dynamic motion plans (a trajectory of joint poses). These plans fulfill different task requirements, i.e., lifting task (requiring "strong" pose) and ballistic task (requiring "fast" pose). Using this formulation, we also propose a method for blending force and velocity requirements to execute the task with a goal lying anywhere between these two extremes. The motion plans generated by

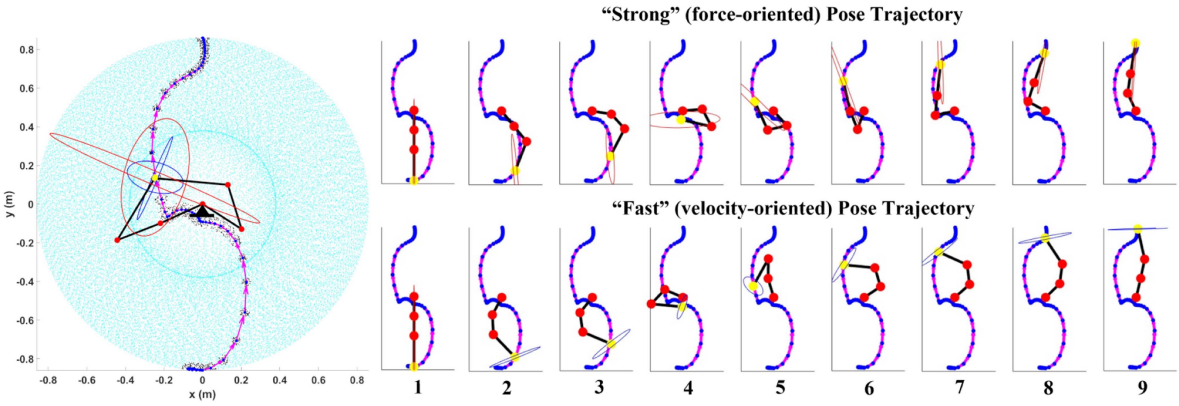

Fig. 2. Kinematic planning based on velocity and force ellipses. (Left) 2D Workspace of the manipulator was sampled (cyan points) and a discrete predefined trajectory was defined (blue dots) with nearest neighbors (black) and motion direction (magenta) marked at each trajectory waypoint. As an example, two pose candidates were generated at a certain waypoint with their velocity (blue) and force (red) ellipses. (Right) The motion sequence of "strong" and "fast" pose trajectories with corresponding ellipses.

the proposed approach are executed experimentally on a torque-controlled, three-link serial planar manipulator to evaluate the pose-enabled benefits in terms of reducing joint torques for lifting task, fully utilizing joint velocity for ballistic task, or somewhere in between for the blended task.

II. Proposed Approach

A. Task-Specific Pose Selection Using Manipulability Metrics

A general motion task can be first characterized by the desired direction of velocity and force (resisting or exerting) application. Some basic tasks might predominantly prioritize application of force over velocity (e.g., weightlifting), while others - generation of a higher velocity of motion (e.g., badminton) with a smaller load, so the joint configuration in a redundant system could be optimized to achieve this single objective. In order to select the suitable manipulator configuration for a specific task, we first introduce Chiu's insights [11] to provide guidance on how to shape and orient the velocity/force ellipses[2] during planning. Specifically, tasks that require large velocity/force should occur in the direction of the major axis of the corresponding ellipse, and those requiring fine velocity/force control should occur along the minor axis (Fig. 1). Let us illustrate these ideas with a purely kinematic example on a three-link planar manipulator used in Fig. 1. The manipulator kinematics is defined in (1), where $\boldsymbol{p} \in \mathbb{R}^2$ is the EE position, $\boldsymbol{q}$ is the state (joint configuration) vector ($\boldsymbol{q} = [\theta_1, \theta_2, \theta_3]^\top$) and θ_i are joint angles.

$$\boldsymbol{p}(\boldsymbol{q}) = \begin{bmatrix} l_1 sin(q_1) + l_2 sin(q_1 + q_2) + l_3 sin(q_1 + q_2 + q_3) \\ -l_1 cos(q_1) - l_2 cos(q_1 + q_2) - l_3 cos(q_1 + q_2 + q_3) \end{bmatrix} \tag{1}$$

In this example of a three-link planar manipulator, different joint pose trajectories can be resolved to be "strong" (with force-priority) or "fast" poses (with velocity-priority), given the same desired workspace trajectory. First, we sample the joint space by iterating through joint angles and use forward kinematics to obtain the corresponding samples in the workspace (1). Given a discrete pre-defined workspace trajectory, 50 nearest neighbors are identified among the samples based on Euclidean distance at each point along the trajectory. These neighbors are the candidate poses for each workspace coordinate (Fig. 2, Left). Among these candidates, depending on the task, best choices of the pose will maximize the radius of velocity/force ellipse along the motion/force direction of interest. Let $\boldsymbol{v}$ denote the unit vector in the target direction (either motion or force direction). α or β denote the ellipse radius (i.e., distance along v from ellipse center to a point on the ellipse) (Fig. 1). They must satisfy (2) for velocity ellipse and (3) for force ellipse [11]:

$$(\alpha \boldsymbol{v})^\top (JJ^\top)^{-1} (\alpha \boldsymbol{v}) = 1 \tag{2}$$

$$(\beta \boldsymbol{v})^\top (JJ^\top) (\beta \boldsymbol{v}) = 1 \tag{3}$$

Therefore, $\alpha = [\boldsymbol{v}^\top (JJ^\top)^{-1} \boldsymbol{v}]^{-1/2}$ and $\beta = [\boldsymbol{v}^\top (JJ^\top) \boldsymbol{v}]^{-1/2}$ giving the radii along directions of interest ($\boldsymbol{v}$) for velocity and force ellipses, respectively.

If we are interested in a **"Strong" Pose Trajectory**, we only consider the direction of the force application. For instance, in Fig. 1 we use a constant vertical gravity vector. Thus the pose selected by the above force criterion should be efficient at supporting a heavy payload. We select these poses iterating through the trajectory, and at each point select the pose whose force ellipse has the largest β along the gravity vector.

In contrast, if we want to construct a **"Fast" Pose Trajectory.**, we only consider the direction of motion, which is a vector pointing from the current workspace trajectory coordinate to the next. Note that direction of this vector varies along the path, unlike the payload force vector in the "strong" case, which is always vertical. In a similar fashion to the "strong" trajectory, for each point we now select the pose whose velocity ellipse has the largest α along the direction of the desired motion.

The motion trajectories resulting from these simple heuristics are quite distinct and intuitive (Fig. 2, Right). The "strong" pose trajectory is similar to a human lifting or upward pushing action. The "fast" pose trajectory is similar to an arm swing pattern, where the third link first winds up to the base and then swings at the target. In both cases, we assume there is a 6 kg payload attached to EE and joint velocities are limited (see Section II-B). To quantify the trajectories, each frame is treated as quasi-static. The workspace velocity capability along

[2]Considering that all examples in this letter, are in 2D, we will refer to these ellipsoids as ellipses from now on.

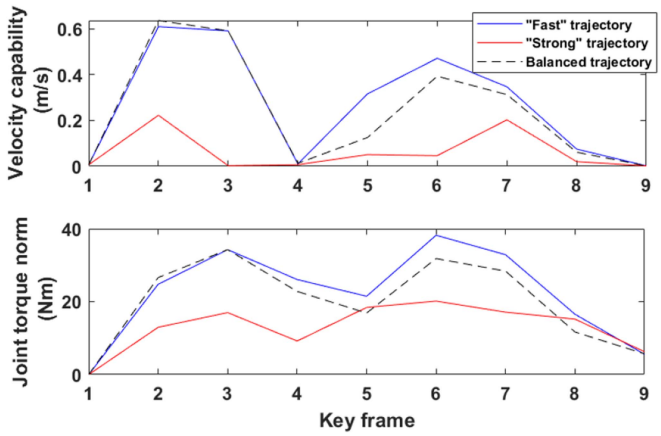

Fig. 3. Joint torque required and workspace velocity potential at each frame for example trajectories.

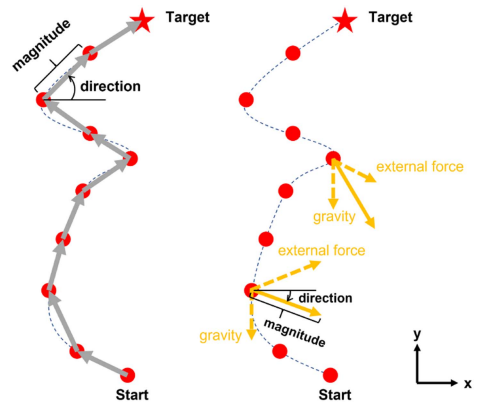

Fig. 4. Definition of desired motion direction (left) and force application direction (right) in trajectory optimization. Red dots indicate the grid points of discrete trajectory expressed in workspace coordinates based on the state vector at current optimization iteration. Blue dashed lines visualize the planned trajectory at current iteration after smoothing.

the desired motion direction can be approximated by mapping the joint velocity limit to workspace by J and projecting onto the motion direction. More accurate calculation can be obtained with manipulability polytope [17]. Similarly, required joint torque to sustain the payload can be calculated via $J^\top$. As shown in Fig. 3, given the same workspace trajectory, "fast" poses have a potential to generate higher linear velocity, while "strong" poses require less joint torque to sustain the load. A balanced trajectory would fall between these two extremes. Note that the "fast" and "strong" trajectories shown here intend to illustrate the difference between the extremes, but are not the "strongest" and the "fastest", so the "balanced" trajectory might slightly violate the bounds. This motivating example suggests that velocity and force ellipses can be useful metrics in reasoning about how humans pick suitable poses for corresponding tasks. Next, we incorporate system dynamics into the planning to generate dynamically feasible trajectories, while keeping this flavor of pose selection.

B. Hardware and Robot Dynamic Model

The 3-DOF planar manipulator used in this study is a custom robot assembly using Series Elastic Actuators (SEA). The first two joints (θ_1 and θ_2) are equipped with larger 400 W brushless motors and the third joint (θ_3) has a smaller 200 W brushless motor. These actuators have an integrated strain wave gearing and a series torsional spring. Larger actuators can produce 65 Nm continuous torque with the maximum velocity of 4 rd/s, while the smaller actuator delivers 36 Nm continuous torque and 4 rd/s velocity, both under 48 V power supply. In the planning phase, considering that near the velocity limit, the actuator performance might be affected by back-EMF, we cap joint velocities at 3 and 5 rd/s, respectively.

The dynamics of this manipulator can be easily derived via Lagrangian method and described by a standard manipulator equation:

$$\boldsymbol{M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = u}, \tag{4}$$

where $\boldsymbol{M(q)}$ is the mass matrix, $\boldsymbol{C(q,\dot{q})}$ is the Coriolis matrix, $\boldsymbol{g(q)}$ is the gravity vector, $\boldsymbol{u}$ is the joint torque vector, and $\boldsymbol{\dot{q}}$ is the joint velocity vector. The effect of reflected inertia, stiction, and viscous friction at joints is minimized by low-level torque control (more details in Section II-E), so all SEAs are modeled as ideal torque sources. For different testing scenarios, we attach 6 and 2 Kg payloads, which we use in (4) before planning.

C. Task-Specific Trajectory Optimization

In this section, we incorporate task-specific pose consideration into system dynamics and formulate a nonlinear trajectory optimization problem (5). Let $\boldsymbol{x}$ denote the state vector, $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]^\top$. Then:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{u},t_0,t_F} \quad & R_t(t_F - t_0) + \int_{t_0}^{t_F} \left(u^\top \boldsymbol{R_u} u + \frac{R_p(\boldsymbol{x},\boldsymbol{u})}{G(\boldsymbol{x})} \right) dt \\
\text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x},\boldsymbol{u}) \\
& \boldsymbol{p(x_0)} = \boldsymbol{p_{initial}} \\
& \boldsymbol{p(x_F)} = \boldsymbol{p_{target}} \\
& [-\pi, 0, 0]^\top \le \boldsymbol{q} \le [\pi, \pi, \pi]^\top \\
& \boldsymbol{\dot{q}_{min}} \le \boldsymbol{\dot{q}} \le \boldsymbol{\dot{q}_{max}} \\
& \boldsymbol{u_{min}} \le \boldsymbol{u} \le \boldsymbol{u_{max}}
\end{aligned} \tag{5}$$

Here, the decision variables are the state vector, control efforts, and both initial and final time of the trajectory. $f(x,u)$ is the nonlinear dynamics constraint based on (4). $G(x)$ is the pose-aware cost weighted by $R_p \in \mathbb{R}$ and it is defined by $\alpha(x)$ or $\beta(x)$ depending on whether the goal is to generate "fast", "strong", or "blended" poses. There are also terms in the cost function to regularize trajectory duration weighted by $R_t \in \mathbb{R}$ and actuator torques weighted by $\boldsymbol{R_u} \in \mathbb{R}^{3\times3}$.

In the context of trajectory optimization, the desired motion or desired force directions are defined as shown in Fig. 4 at the grid points of the trajectory. Similarly to the previous kinematic planning example, motion direction is defined as a workspace vector pointing from the current grid point to the next, and the force application direction is determined by the vector sum of vectors of gravity and external force. Often, external force is unknown during pre-planning, but if it could be modeled based on experience, or sensed if implemented in a real-time controller, then the external force can be explicitly considered in

the trajectory optimization. In this work, we assume the external force to be zero, so the force direction is governed by gravity and always points down.

Furthermore, the weight of the selected pose, R_p, will be a function of the current state and control efforts and are different at each grid point, for each trajectory optimization iteration. For tasks requiring "strong" poses, $R_p = R_{pf} u^T u$, i.e., if the torque planned by the optimizer is large at this grid point, more weight will be placed on selecting a strong pose. Similarly, for tasks requiring "fast" poses, $R_p = R_{pv} \dot{q}^T \dot{q}$, meaning that more weight is placed on the fast pose when the planned instantaneous joint velocity is high. R_{pf} and R_{pv} are constant weights $\in \mathbb{R}$. We refer to this dynamic weighting heuristic of pose selection as "request strong/fast pose on demand".

We then formulate trajectory optimization objectives as to obtain 1) a trajectory with "strong" poses against certain heavy payload using as little joint torque as possible to reach the target position in workspace (similar to a weightlifting motion):

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u},t_0,t_F} \quad & R_t(t_F - t_0) + \int_{t_0}^{t_F} \left(u^\top \boldsymbol{R_u} u + \frac{R_{pf} \bar{\boldsymbol{u}}^\top \bar{\boldsymbol{u}}}{\beta(x)} \right) dt \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \\ & \boldsymbol{p}(\boldsymbol{x_0}) = \boldsymbol{p_{initial}} \\ & \boldsymbol{p}(\boldsymbol{x_F}) = \boldsymbol{p_{target}} \\ & \dot{\boldsymbol{q}}(t_0) = \dot{\boldsymbol{q}}(t_F) = [0,0,0]^\top \end{aligned} \tag{6}$$

or 2) a trajectory with "fast" poses, so that the manipulator's EE reaches the maximum linear velocity at the end of the trajectory, given the limited joint velocity (similar to a ballistic motion):[3]

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u},t_0,t_F} \quad & R_t(t_F - t_0) + \frac{R_v}{\dot{p}(\boldsymbol{x_F})^\top \dot{p}(\boldsymbol{x_F})} \\ & + \int_{t_0}^{t_F} \left(u^\top \boldsymbol{R_u} u + \frac{R_{pv} \dot{\bar{\boldsymbol{q}}}^\top \dot{\bar{\boldsymbol{q}}}}{\alpha(\boldsymbol{x})} \right) dt \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \\ & \boldsymbol{p}(\boldsymbol{x_0}) = \boldsymbol{p_{initial}} \\ & \dot{\boldsymbol{p}}_{\boldsymbol{x}}(\boldsymbol{x_F}) = \dot{\boldsymbol{p}}_{\boldsymbol{y}}(\boldsymbol{x_F}) > 0 \end{aligned} \tag{7}$$

All other constraints on joint range of motion, velocity, and torque still hold. $\dot{\bar{\boldsymbol{q}}}$ and $\bar{\boldsymbol{u}}$ are normalized by joint torque and velocity limit.

All trajectories are transcribed with a 20-point grids using the direct collocation method [6] and optimized using OptimTraj package [18] in MATLAB (R2023b) with two passes. The first pass uses an interior-point solver with the initial guess being a linear interpolation between $\boldsymbol{q} = [0,0,0]^\top$ (hanging down) and $\boldsymbol{q} = [\pi,0,0]^\top$ (upright), while $\dot{\boldsymbol{q}}$ and $\boldsymbol{u}$ are initialized to zero. The second pass warm-starts with the first pass result as initial guess, using a sequential quadratic programming solver. OptimTraj can usually find a solution in 10-15 s. If real-time application is of interest in the future, CasADi [19] will be a more suitable package and a test implementation shows the computation time is around 2 s.

[3]Note that in this problem formulation (7), the optimal ballistic direction is assumed to be 45°, so EE's linear velocities are constrained to be the same in x and y directions.

TABLE I
SUMMARY OF PARAMETERS IN TRAJECTORY OPTIMIZATION

	R_t	R_u	R_v	R_{pf}	R_{pv}
"Strong" Pose Trajectory			-	50	-
"Fast" Pose Trajectory	5	diag([5,5,5])	1e6	-	50
Blend Pose Trajectory			-	1000	1000

D. Pose Blending

More complex tasks may require a combination of different types of poses for both velocity and force. A natural way to blend pose costs is to put $\frac{R_{pv} \dot{\bar{\boldsymbol{q}}}^\top \dot{\bar{\boldsymbol{q}}}}{\alpha(\boldsymbol{x})}$ and $\frac{R_{pf} \bar{\boldsymbol{u}}^\top \bar{\boldsymbol{u}}}{\beta(\boldsymbol{x})}$ in the same cost function. In this case R_{pv} and R_{pf} will determine the relative importance between "strong" and "fast" poses. However, α and β have very different magnitudes, since velocity and force ellipses are reciprocal to each other. One simple way to obtain the relative ellipse radius ($\bar{\alpha}$ and $\bar{\beta}$) is by re-using the previously obtained workspace samples, finding the k nearest neighbors of each grid point (in this case, $k = 50$), and dividing by the maximum ellipse radius found among the neighbors (including itself) (8). This is be done for each grid point at each optimization iteration. It is assumed the task is to move from $\boldsymbol{p_{initial}}$ to $\boldsymbol{p_{target}}$ with a 6 kg payload, similar to "strong" pose trajectory. All these quantities are between 0 and 1, so, correspondingly, R_{pv} and R_{pf} will have much larger values (Table I). The equal weights in this case indicate a balance of "strong" and "fast" poses, but if one type of pose is prioritized, the weights can be adjusted accordingly.

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u},t_0,t_F} \quad & R_t(t_F - t_0) \\ & + \int_{t_0}^{t_F} \left(u^\top \boldsymbol{R_u} u + \frac{R_{pv} \dot{\bar{\boldsymbol{q}}}^\top \dot{\bar{\boldsymbol{q}}}}{\bar{\alpha}(x)} + \frac{R_{pf} \bar{\boldsymbol{u}}^\top \bar{\boldsymbol{u}}}{\bar{\beta}(x)} \right) dt \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \\ & \boldsymbol{p}(\boldsymbol{x_0}) = \boldsymbol{p_{initial}} \\ & \boldsymbol{p}(\boldsymbol{x_F}) = \boldsymbol{p_{target}} \\ & \dot{\boldsymbol{q}}(t_0) = \dot{\boldsymbol{q}}(t_F) = [0,0,0]^\top \end{aligned} \tag{8}$$

E. Low-Level Control

A motion plan generated by the trajectory optimization provides feedforward torque reference ($\boldsymbol{\tau_{ref}}$) and motion reference ($\boldsymbol{x_{ref}}$). These references are tracked with a time-varying linear quadratic regulator (LQR) (9). This feedback controller linearizes the nonlinear manipulator (4) along the reference motion trajectory and calculates the LQR gain for each linearized dynamic model ($\boldsymbol{K(t)}$). Reference torque is added to the feedback torque command in a feedforward fashion to obtain the total torque command (τ_{cmd}).

$$\boldsymbol{\tau_{cmd}} = \boldsymbol{\tau_{ref}}(t) + \boldsymbol{K(t)}(\boldsymbol{x} - \boldsymbol{x_{ref}}(t)) \tag{9}$$

The torque command is executed at each SEA joint with a low-level torque controller. The torque controller is a PD controller with a disturbance observer with rejection frequency of 30 Hz [20]. Regardless of the high-level command, each SEA always renders a zero torque control mode at the joint to remove most of the unmodeled dynamics (stiction, viscous friction, etc.), so that the high-level trajectory optimization can use an

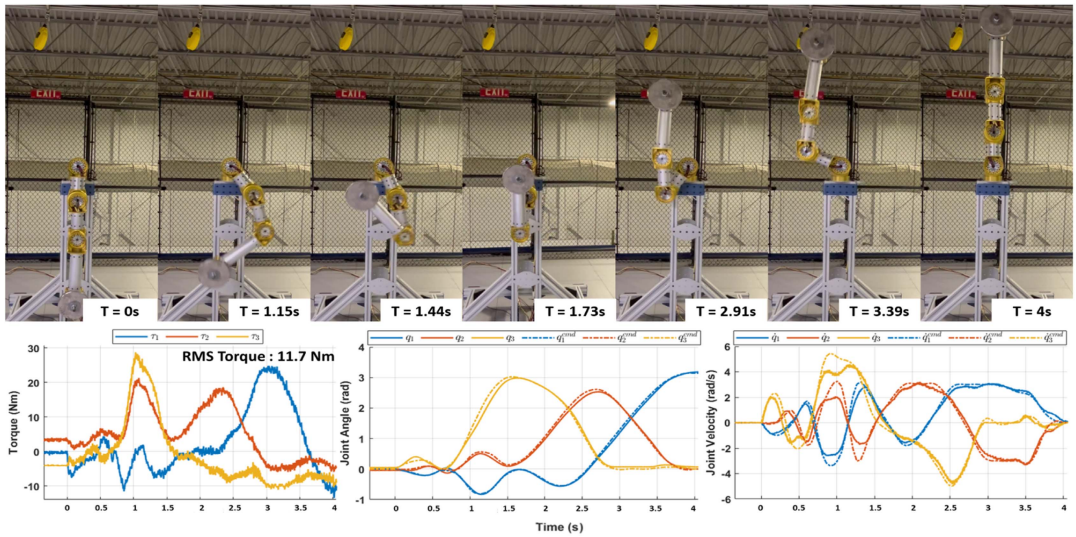

Fig. 5. "Strong" pose trajectory with measured joint torque and tracking of joint angle and velocity.

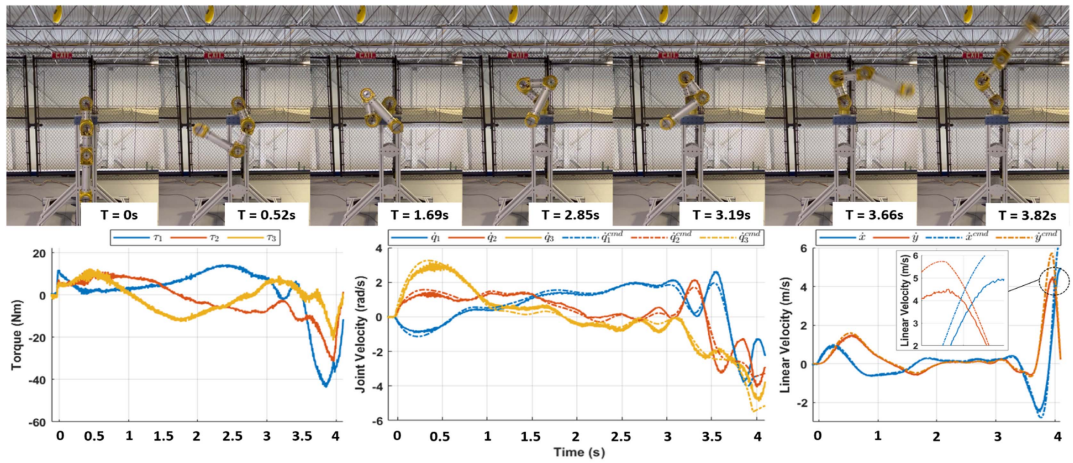

Fig. 6. "Fast" pose trajectory with measured joint torque, as well as tracking of joint velocity and EE's linear velocity.

ideal dynamic model (4). During the execution of a motion plan, the torque commanded to each SEA is the sum of zero torque control efforts and high-level torque command (9). All communication and controller computation are implemented in MATLAB Simulink Real-Time and deployed on a real-time embedded computer (Speedgoat) at 1 kHz.

III. Experiment

A. "Strong" Pose Trajectory

In this experiment, the three-link manipulator tries to reach to an upright target point in the workspace, while carrying a heavy payload (6 kg). Since the force direction is set to be aligned with the gravity, the pose selection will prefer the manipulator poses that are efficient at resisting the downward force. As a result, the planned motion goes through two intermediate singularities, i.e., third link pointing up and then second link pointing up, while performing brief rapid motions to move between singularities until eventually reaching the target (Fig. 5). This motion sequence is similar to the weightlifting "clean and jerk" technique. Near $t = 1$s, where large joint torque and velocity are requested simultaneously to pull the payload up (Frame 2 and 3 in Fig. 5), joint velocities deviate from the planned trajectories, possibly due to the rise of back EMF.

B. "Fast" Pose Trajectory

In this experiment the objective is to generate maximum linear velocity at the manipulator's EE, given limited joint velocity and a lighter payload (2 kg). In this case, the manipulator has no definitive target coordinates at the end of the trajectory and it is assumed the final resulting linear velocity would point at 45° upward from the horizontal plane. The planned motion initially goes through a series of arm "wind-ups", and then rapidly actuates the joints to extend the arm and reach an EE linear velocity of about 4.4 m/s (Fig. 6). This motion sequence is quite similar to the badminton "high clear" shot motion, if

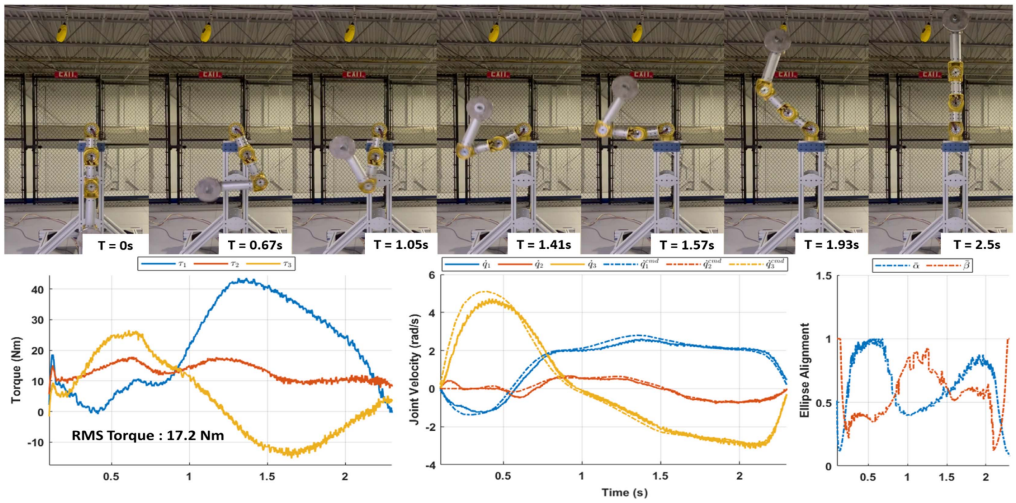

Fig. 7. Blended pose trajectory with measured joint torque and tracking of joint velocity, as well as ellipse alignments.

one imagines the first two links are upper arm and forearm, and the third link is the racket. In this experiment, the actual actuator torque undershoots the command towards the end of the trajectory, so the velocity tracking is degraded. When the linear velocity points at the 45° direction, the x and y magnitudes are approximately 3.5 m/s.

C. Blended Pose Trajectory

In the last experiment, the manipulator again tries to reach the upright target with a 6 kg payload, while this time both velocity and force ellipses are considered during planning to achieve a trajectory with a balance of "strong" and "fast" poses. According to the planned motion, the manipulator first quickly swings up the third link to move the payload near the base and then pushes the weight up to the target in the second stage of the motion (Fig. 7). This motion is similar to the combination of the two trajectories in the kinematic example (Fig. 2). As indicated by ellipse alignments, the motion initiates with "fast" pose motion (Frame 1-2), followed by a "strong" pose motion (Frame 3), and ends with a mixed pose motion (biased to "fast") (Frame 4-7). Compared to the "strong" pose trajectory where only force ellipse is considered (Fig. 5), the main difference is how the payload is handled after it gets pulled close to the base (starting from Frame 4 in Fig. 7). In the blend case, the manipulator does not intentionally seek singularities and does not linger near the base, to achieve a faster motion (2.3 vs. 4 s) at the cost of increased joint torques, i.e., root mean square (RMS): 17.2 vs 11.7 Nm.

IV. Discussion

The work presented in this paper is motivated by the desire to driving down the robot size by purposefully planning the trajectory to push the robot's performance to the extreme in terms of velocity and force. An approach to formulating task-specific motion planning is proposed and several motions with different goals are generated. Interestingly, by explicitly taking force ellipse into account, the "strong" pose trajectory proactively includes singular configurations along the trajectory so that the payload is sustained by robot structure for most of the time and actuator torque is mainly required to support the arm to move to the next singularity (Fig. 5). The peak torque is approximately 27 Nm (at τ_1) to lift a 6 kg payload, as opposed to a naive trajectory that extends the arm straight horizontally would take roughly a peak torque of 51 Nm (moment arm of 0.86 m and payload of 6 kg, without considering the robot's own weight of 7 kg). On the other hand, "fast" pose trajectory is a less surprising, trebuchet-like ballistic motion. The planned linear velocity approached the theoretical maximum velocity of 4.5 m/s. However, it is worth noting that this was not the only motion sequence that could generate a high EE linear velocity. For example, another possible motion found during investigation is completing the trajectory from the bottom, similar to a forehand serve motion in badminton. This pose variation is likely due to the weight tuning in the optimization.

We found the formulation of the trajectory optimization proposed in this work to be quite flexible for integrating system dynamics, pose-aware costs, and constraints. However, there are a few limitations in this formulation. First, the desired motion direction is currently defined between adjacent grid points (Fig. 4), but another reasonable approach would be to point from each grid point directly to the target. The former is chosen because there is no target defined for the "fast" pose trajectory. However, the advantage of the latter is to reduce the dependency between decision variables (i.e., states). Furthermore, only the direction information is utilized (Fig. 4) in guiding the planned orientation of the ellipses, but magnitude information can also be helpful in terms of determining the relative weights between grid points or between velocity/force priorities.

While the meaning of pose-aware weights (R_{pv} and R_{pf}) is intuitive (8), these weights should be balanced with the regularization weights, and their tuning is not straight-forward. For example, if R_{pv} and R_{pf} are too large, the planned motion can intentionally take a long-duration trajectory so that the manipulator can stay at the desired pose for a longer time to reduce cost. On the other hand, if we remove the pose-aware cost terms in favor of increasing R_u or R_v for "strong" or "fast" trajectory, the feasible trajectories can still be found, but are far less intuitive and might degrade with the increased number of DOF. In this sense, one benefit of including the pose-aware cost term is to help guide the optimizer to find the more interpretable pose solutions.

The proposed task-specific trajectory optimization formulation provides an intuitive and versatile approach to account for the task requirement and pose advantages for various tasks. It is also fairly straight-forward to extend this approach to different stages of a single task, i.e., different pose weights for each stage based on force/velocity priority. In addition, although in this work we assume the gravitational force of the payload is the dominating external force, in reality, there could be other non-negligible external forces. One way to extend the proposed formulation in those scenarios is to regularize the minor axis of force ellipse (i.e., the smallest eigenvalue of $JJ^\top$). This approach will force the solution to stay close to singularities but with some safety margin. Thus, we can still exploit the force capability of near-singular configurations, but also not lose the ability to respond to disturbances in other directions.

Finally, in the scope of this work, we only exploit the alignment of the manipulator's force and velocity ellipses with the task in terms of generating large velocity or large force. However, generating trajectories with high-resolution velocity or force capability can be another potential application of the proposed approach and will be important for precise manipulation tasks such as robotic assembly, and human-robot interaction. Furthermore, this strategy can be implemented within a real-time model-predictive control framework. In that case the measured external force and state updates can provide more accurate dynamic guidance on force and velocity ellipse selection, regulating pose weights in real-time. We leave theses directions for our future work.

V. Conclusion

In this letter, we explored the approach for a redundant robot manipulator to exploit the benefits of different poses for a given a task. We encoded this task-specific pose choice into a trajectory optimization formulation, and introduced a method for blending force- and velocity-oriented pose selection. The proposed approach is promising in generating task-optimized and interpretable motion plans. Additionally, the formulation of this method can be generalized to pre-planning of more complex motions, as well as a real-time implementation.

Acknowledgment

The authors would like to thank Jessica Leu and Mahyar Abdeetedal for their insights and suggestions; Sean Cai, Patryk Mordec, Sean Garcen, and Jan Borkowski for hardware support.

References

[1] A. Goldenberg, B. Benhabib, and R. Fenton, "A complete generalized solution to the inverse kinematics of robots," *IEEE J. Robot. Automat.*, vol. 1, no. 1, pp. 14–20, Mar. 1985. [Online]. Available: https://ieeexplore.ieee.org/document/1086995/

[2] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man Mach. Syst.*, vol. 10, no. 2, pp. 47–53, Jun. 1969. [Online]. Available: https://ieeexplore.ieee.org/document/4081862/

[3] A. Dietrich, C. Ott, and A. Albu-Schäffer, "An overview of null space projections for redundant, torque-controlled robots," *Int. J. Robot. Res.*, vol. 34, no. 11, pp. 1385–1400, Sep. 2015. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0278364914566516

[4] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001. [Online]. Available: http://journals.sagepub.com/doi/10.1177/02783640122067453

[5] C. Hargraves and S. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *J. Guid. Control Dyn.*, vol. 10, no. 4, pp. 338–342, Jul. 1987. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.20223

[6] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming* (Series Advances in Design and Control), 2nd ed. Philadelphia, PA, USA: SIAM, 2010. [Online]. Available: https://epubs.siam.org/doi/book/10.1137/1.9780898718577

[7] B. Katz, J. D. Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6295–6301. [Online]. Available: https://ieeexplore.ieee.org/document/8793865/

[8] M. Posa, S. Kuindersma, and R. Tedrake, "Optimization and stabilization of trajectories for constrained dynamical systems," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1366–1373. [Online]. Available: https://ieeexplore.ieee.org/document/7487270/

[9] T. Yoshikawa, "Manipulability and redundancy control of robotic mechanisms," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1985, pp. 1004–1009. [Online]. Available: https://ieeexplore.ieee.org/document/1087283/

[10] J. Haviland and P. Corke, "NEO: A novel expeditious optimisation algorithm for reactive motion control of manipulators," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1043–1050, Apr. 2021.

[11] S. Chiu, "Control of redundant manipulators for task compatibility," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1987, pp. 1718–1724. [Online]. Available: https://ieeexplore.ieee.org/document/1087795/

[12] S. Lee, "Dual redundant arm configuration optimization with task-oriented dual arm manipulability," *IEEE Trans. Robot. Automat.*, vol. 5, no. 1, pp. 78–97, Feb. 1989. [Online]. Available: https://ieeexplore.ieee.org/document/88020/

[13] B.-H. Jun, P.-M. Lee, and J. Lee, "Manipulability analysis of underwater robotic arms on ROV and application to task-oriented joint configuration," in *Proc. Oceans MTS/IEEE Techno-Ocean*, 2004, pp. 1548–1553. [Online]. Available: https://ieeexplore.ieee.org/document/1406352/

[14] I. Lee and J.-H. Oh, "Humanoid posture selection for reaching motion and a cooperative balancing controller," *J. Intell. Robot. Syst.*, vol. 81, no. 3/4, pp. 301–316, Mar. 2016. [Online]. Available: http://link.springer.com/10.1007/s10846-015-0225-z

[15] N. Jaquier, L. Rozo, D. G. Caldwell, and S. Calinon, "Geometry-aware manipulability learning, tracking, and transfer," *Int. J. Robot. Res.*, vol. 40, no. 2/3, pp. 624–650, Feb. 2021. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0278364920946815

[16] B. J. Martin and J. E. Bobrow, "Minimum-effort motions for open-chain manipulators with task-dependent end-effector constraints," *Int. J. Robot. Res.*, vol. 18, no. 2, pp. 213–224, Feb. 1999. [Online]. Available: http://journals.sagepub.com/doi/10.1177/027836499901800206

[17] J. Lee, "A study on the manipulability measures for robot manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. Innov. Robot. Real-World Appl.*, 1997, pp. 1458–1465. [Online]. Available: https://ieeexplore.ieee.org/document/656551/

[18] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Rev.*, vol. 59, no. 4, pp. 849–904, Jan. 2017. [Online]. Available: https://epubs.siam.org/doi/10.1137/16M1062569

[19] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, Mar. 2019. [Online]. Available: http://link.springer.com/10.1007/s12532-018-0139-4

[20] N. Paine et al., "Actuator control for the NASA-JSC Valkyrie humanoid robot: A decoupled dynamics approach for torque control of series elastic robots," *J. Field Robot.*, vol. 32, no. 3, pp. 378–396, May 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/rob.21556