2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

A Time-Optimal Energy Planner for Safe Human-Robot Collaboration

Andrea Pupa[†,1], Marco Minelli[†,1] and Cristian Secchi[1]

***Abstract*— The human-robot collaboration scenarios are characterized by the presence of human operators and robots that work in close contact with each other. As a consequence, the safety regulations have been updated in order to provide guidelines on how to asses safety in these new scenarios. In particular, Power and Force Limiting (PFL) collaborative mode describes how the energy should be regulated during the collaboration. Based on these guidelines, we propose a new optimal trajectory planner which, by exploiting the variability of the robot's inertia as a function of its configuration, is able to return trajectories that can be travelled at greater speed and in less time, while guaranteeing the safety limits according to the standard. The proposed planner was validated first in simulation, comparing completion times with other state-of-the-art planning algorithms, and then experimentally, demonstrating the performance of the planned trajectories during physical interaction with the environment. Both validations confirm the effectiveness of the proposed planner, which returns shorter completion times while ensuring safe interaction.**

I. Introduction

Ensuring safety in human-robot collaboration (HRC) scenarios is of pivotal importance [1]. The ISO 10218-1 and the ISO 10218-2 [2], [3] standards codify four different collaborative modes: *safety-rated monitored stop* (SMS), *hand guiding* (HG), *speed and separation monitoring* (SSM) and *power and force limiting* (PFL). For each collaborative mode, the technical specification ISO/TS 15066 [4] provides guidelines on the risk assessment. PFL, collaborative mode in which the motor power and force are limited, is of particular importance when a a side-by-side collaboration is required.

A lot of research has been done in this field, addressing safety both at the planning level [5]–[7] and at the control level [8]–[10]. In particular, among the latter, some works explicitly take into account safety standards to improve collaboration. In [11] the PFL constraint is addressed at a control level with an energy tank-based framework. This approach, however, does not guarantee the global optimality of the implemented behavior. In [12] the authors propose a safety kinodynamic framework that allows to improve the performances of the collaboration while ensuring that the robot behaviour is compliant with the SSM velocity limit. In [13] a trajectory scaling algorithm that ensures a safe robot behaviour is proposed. In particular, the velocity limits imposed by SSM and PFL are merged in order to define a new safe velocity limit and used as an upper bound in the trajectory scaling algorithm. Similarly, in [14] the authors exploit control barrier functions (CBFs) to ensure that the final robot velocity is lower than the limit derived by combining both the SSM and the PFL.

The problem with these approaches is that they require monitoring the human operator at runtime. Thus, they are affected by all the difficulties regarding the monitoring system, such as occlusions. Furthermore, none of these approaches takes into account the aspect of safety certification. In industrial settings where monitoring systems are used for safety-critical purposes, adherence to safety standards and certification procedures is mandatory. Failure to meet these standards could compromise the overall safety integrity of the system. Therefore, these methods are unsuitable for real industrial scenarios due to their reliance on standard non-safe sensors, such as the optitrack system [12] or stereocameras [14].

To avoid the problems related to the monitoring system, it is better to address safety at the planning level. In fact, a proper planning strategy makes it possible to offline generate trajectories that take into account the presence of the human operator, e.g. limiting the velocity of the end-effector [15].

In [16], the rapidly-exploring random tree (RRT) algorithm is introduced, while in [17], [18], two variants are proposed, RRT-Connect and RRT. In [19], the authors introduce CHOMP algorithm, which continuously refines an initial path by solving an optimization problem balancing smoothness and obstacle avoidance. Based on CHOMP, [20] presents STOMP, an algorithm that explores around the initial trajectory to generate a lower-cost final one, even for non-differentiable or non-smooth cost functions.

The problem with these planners is that they are parametrization invariant, meaning they can only yield an optimal path and not an optimal trajectory. Indeed, they are not able to address time optimality in the presence of velocity constraints, such as the ones of the safety regulation. Furthermore, when a safe-certified monitoring system is not available, the only way to ensure safety is to ensure that every possible collision is compliant with the safety standards.

Regardless of which collaborative mode is adhered to, the ISO guidelines provide a maximum speed limit, which apparently limits the minimum execution time of a trajectory to the ratio of the length of the trajectory to the maximum applicable speed. However, it is possible to overcome these limits by considering the problem directly from the energetic point of view.

This work proposes to address the ISO/TS 15066 constraint at the planning level. Since the energy of the robot is

† *Andrea Pupa and Marco Minelli contributed equally to this work.*

[1] Andrea Pupa, Marco Minelli and Cristian Secchi are with the Department of Science and Methods of Engineering, University of Modena and Reggio Emilia, Italy. E-mail: {andrea.pupa, marco.minelli, cristian.secchi}@unimore.it,

Project funded under the National Recovery and Resilience Plan (NRRP), Mission 04 Component 2 Investment 1.5 – NextGenerationEU, Call for tender n. 3277 dated 30/12/2021. Award Number: 0001052 dated 23/06/2022.

directly proportional to the inertia matrix and to the square of its velocity, it is possible to exploit the variability of the inertia matrix with respect to the configuration to find trajectories that exhibit the same energy but that can be performed at a different speed. This means that by decreasing the inertia matrix it is possible to increase the speed and decrease the execution time. Increasing only the speed, in fact, does not make possible, a priori, to decrease the execution time of the trajectory, as it could mean travelling a longer distance to comply with the energetic bound.

In order to find the best solution, the proposed planning strategy is based on the solution of an optimization problem, which allows obtaining time-optimal trajectories while guaranteeing the energy limit imposed by the ISO.

Given the high non-linearity of the considered problem, mainly introduced by the explicit consideration of the inertia matrix of the manipulator and the non-collision constraints, the solution of the problem in a single stage is not effective both from the temporal point of view and from the quality of the solution obtained. For this reason, it was decided to split the solution to the problem into several stages.

The main contributions of this paper are:

- An energy-based motion planner that returns safe and time-optimal trajectory that can be used to implement collaborative tasks when an external tracking system is not available
- A comparison of the performance between the proposed planner and state-of-the-art trajectory planners
- An experimental validation of the performance of the proposed planner when collisions occur during the trajectory execution.

The rest of the paper is organized as follows: in Sec. II the planning problem with attention to the ISO/TS 15066 is detailed. Sec. III describes the PFL collaborative mode guidelines. Sec. IV and V report the formalization of the optimization problem and the strategy implemented to solve it, respectively. The validation of the proposed planner is reported Sec. VI-A and VI-B while Sec. VII addresses the conclusions and future works.

II. Problem Statement

Consider an HRC application where a human operator and n-DoFs velocity-controlled manipulator have to collaborate in order to accomplish a job. The robot can be modeled as:

$$\dot{\mathbf{q}} = \mathbf{u}, \tag{1}$$

where $\dot{\mathbf{q}} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^n$ are the joint velocities and the controller input, respectively.

While performing the job, the robot has to continuously move from the actual initial configuration $\mathbf{q}(t_i) = \mathbf{q}_i \in \mathbb{R}^n$ to a desired final configuration $\mathbf{q}(t_f) = \mathbf{q}_f \in \mathbb{R}^n$. The aim of this work is to design a trajectory planning strategy that returns a proper safe trajectory $\mathbf{q}(\cdot)$ that is time-optimal and that is compliant with the safety guidelines imposed by the ISO/TS 15066, i.e. a trajectory that will not harm the operator in case of collisions. In particular, the PFL is the only collaborative mode that admits the possibility of collisions. For this reason, in this work, the trajectory is considered safe if and only if it is compliant with the PFL limits, i.e. the maximum energy transferred to the human is limited.

III. Power and Force Limiting

In modern industrial applications, the collaborative mode PFL is of paramount importance. In fact, this collaborative mode is one that allows for side-by-side collaboration between the human operator and the robot, where the collision can happen either intentionally or unintentionally. This is ensured by limiting, as the name suggests, the power and force of the robot manipulator. This way, in the event of a collision between the operator and the robot, the energy exchange will not result in harm to the human operator.

In particular, ISO/TS 15066 defines two different types of collisions. *Quasi-static contact*, situations in which part of the human body is trapped between the robot system and another part of the world, e.g. clamping or crushing. *Transient contact*, situations in which part of the human body is impacted by the robot without clamping or trapping, thus the contact has a short duration.

For each type of collision, the technical specification provides a set of thresholds regarding both the maximum admissible force F_{max} and the maximum admissible pressure p_{max}. These thresholds strictly depend on the human body part involved in the collision, which is assumed to be known. While this is a strong assumption in the regulation, it is important to underline that the specific human body parts with a high risk of collision can be estimated with a proper risk assessment evaluation, as detailed in [4]. Then, with the worst-case scenario approach it is possible to find F_{max}.

Starting from these thresholds, it is possible to compute, for each body part, the maximum transferred energy E_{max}. In particular:

$$E_{max} = \frac{F_{max}^2}{2k}, \tag{2}$$

where k is the effective spring constant of the specific body part and can be found in the technical specification.

Once E_{max} is defined, the regulation [4] proposes a method to evaluate the maximum admissible velocity of the robot. However, this procedure is highly conservative and it is possible to increase the performance by working directly on the energy limits. To achieve this, the contact is modeled as a perfectly inelastic collision, i.e. after the contact the robot and the human body part will have the same velocity, which is the worst scenario that may occur. As showed in [8], the total energy loss is defined as:

$$\Delta\mathbf{K}_e = \frac{1}{2}\frac{m_{R_n} m_H}{m_{R_n} + m_H}\|\dot{\mathbf{x}}_{R_n} - \dot{\mathbf{x}}_{H_n}\|^2, \tag{3}$$

where $m_{R_n} \in \mathbb{R}$ is the apparent mass of the robot, i.e. the mass perceived by the human operator during the collision, and $m_H \in \mathbb{R}$ is the mass of the human body part, which is defined in the ISO/TS 15066. The terms $\dot{\mathbf{x}}_{R_n} \in \mathbb{R}^3$ and $\dot{\mathbf{x}}_{H_n} \in \mathbb{R}^3$ are the cartesian velocity of the robot towards the human operator and vice-versa, respectively. In particular,

both m_{R_n} and $\dot{\mathbf{x}}_{R_n}$ can be computed exploiting the modified Jacobian $J_r(\mathbf{q}) \in \mathbb{R}^{1\times n}$ defined in [12], i.e. the Jacobian that allows taking into account only the scalar velocity towards the human operator, which is defined as:

$$J_r(\mathbf{q}) = \hat{\mathbf{n}}^T J(\mathbf{q}), \tag{4}$$

where $\hat{\mathbf{n}} = \{n_x, n_y, n_z, 0, 0, 0\}$ is the unit vector representing the robot-human direction. This yields to:

$$m_{R_n} = J_r(\mathbf{q})^{-T} M(\mathbf{q}) J_r(\mathbf{q})^{-1}, \tag{5}$$

$$\dot{\mathbf{x}}_{R_n} = J_r(\mathbf{q})\dot{\mathbf{q}}, \tag{6}$$

where $M(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the inertia matrix of the robot and can be computed exploiting the Recursive Newton-Euler algorithm [21]. Merging (2) and (3) it is possible to define the PFL constraint:

$$\Delta\mathbf{K}_e \leq E_{max}. \tag{7}$$

Thus, ensuring (7) all over the entire trajectory $\mathbf{q}(\cdot)$ means that the exchange of energy between the human operator and the robot will not result in harm for the human, making the robot safe. Furthermore, as previously detailed, monitoring the human operator for safe operation requires a 3D safe-certified system, which is in general not available. This means that the unit vector $\hat{\mathbf{n}}$ is not known a priori, and ensuring (7) is not a straightforward operation. However, it is possible to use the worst-case scenario that may arise: the robot is always moving toward the human operator. This leads to the following formulation of the PFL constraint:

$$\begin{cases} \Delta\mathbf{K}_e = \dfrac{1}{2}\dfrac{m_{R_n} m_H}{m_{R_n} + m_H}\|\dot{\mathbf{x}}_{R_n} - \dot{\mathbf{x}}_{H_n}\|^2, \\ m_{R_n} = J_{r,wc}(\mathbf{q})^{-T} M(\mathbf{q}) J_{r,wc}(\mathbf{q})^{-1}, \\ \dot{\mathbf{x}}_{R_n} = J_{r,wc}(\mathbf{q})\dot{\mathbf{q}}, \\ J_{r,wc}(\mathbf{q}) = \hat{\mathbf{n}}_{traj}(\mathbf{q})^T J(\mathbf{q}), \end{cases} \tag{8}$$

where $\hat{\mathbf{n}}_{traj}(\mathbf{q})$ is the unit vector tangent to the trajectory at configuration $\mathbf{q}$ and $J_{r,wc}(\mathbf{q})$ is the Jacobian that allows extracting the linear velocity in the direction of the trajectory, i.e. the magnitude of the linear trajectory. Lastly, $\dot{\mathbf{x}}_{H_n}$ is considered constant and always towards the robot as suggested by the ISO/TS 15066.

IV. Planner modeling

As reported in Section II, the problem of generating a trajectory that moves the robot from an initial configuration $\mathbf{q}_i$ to a desired configuration $\mathbf{q}_f$ is considered. The generated trajectory must be such that the kinetic energy is compliant with the PFL limits, in order to not harm the operator in case of collision.

A. Trajectory parametrization

In order to set up the optimization problem, the trajectory is first parametrized. In this work it has been decided to describe the path as an n-dimensional natural cubic spline $\mathcal{P}(s)$ with: 1) initial value coincident with the initial configuration of the robot $\mathbf{q}_i$, 2) final value coincident with the desired configuration of the robot $\mathbf{q}_f$, 3) and passing through an intermediate parametric configuration $\mathbf{q}_{ref} \in \mathbb{R}^n$, with $s \in [0, L_\mathcal{P}] \subset \mathbb{R}^+$ the path arc length parameter and $L_\mathcal{P}$ the length of the path $\mathcal{P}$. In this way, once the value of the parameter $\mathbf{q}_{ref}$ has been chosen, the path of the trajectory is defined.

When the path is defined, to obtain the trajectory it is necessary to impose the velocity profile with which to travel the path. To ensure smooth execution of the trajectory without affecting the execution time, it has been decided to use a trapezoidal velocity profile, which is typically used within the manipulators' internal planner. This can be achieved by setting:

$$s(t) = \begin{cases} \frac{1}{2}at^2 & 0 \leq t \leq t_c \\ at_c(t - \frac{t_c}{2}) & t_c < t \leq t_c - T \\ L_\mathcal{P} - \frac{1}{2}a(T-t)^2 & T - t_c < t \leq T \end{cases} \tag{9}$$

with:

$$t_c = \frac{T}{2} - \frac{1}{2}\sqrt{\frac{T^2 a - 4L_\mathcal{P}}{a}}, \quad a \geq \frac{4L_\mathcal{P}}{T^2} \tag{10}$$

where $a \in \mathbb{R}^+$ is the velocity profile acceleration, $T \in \mathbb{R}^+$ is the velocity profile duration, i.e. the time the trajectory requires to be performed, and $t_c \in \mathbb{R}^+$ the velocity profile acceleration and deceleration time. This means that, once the value of the parameters a and T has been chosen, the velocity profile of the trajectory is defined.

Finally, the set of parameters used to parameterize the trajectory are:

$$\mathcal{M} = \{\mathbf{q}_{ref}, a, T\} \tag{11}$$

B. Constraints

Four types of constraints are considered in the formulation of the optimization problem:

- *Speed limit*: The velocities of the robot are physically limited. This can be translated into a bound on the trajectory velocities:

$$-\dot{\mathbf{q}}_{max} \leq \dot{\mathbf{q}}(\cdot) \leq \dot{\mathbf{q}}_{max}, \tag{12}$$

  where $\dot{\mathbf{q}}_{max} \in \mathbb{R}^{n+}$ is the robot maximum joint speed vector.
- *Acceleration limit*: The accelerations of the robot are also physically limited. This can be translated into a bound on the trajectory accelerations:

$$-\ddot{\mathbf{q}}_{max} \leq \ddot{\mathbf{q}}(\cdot) \leq \ddot{\mathbf{q}}_{max} \tag{13}$$

  where $\ddot{\mathbf{q}}_{max} \in \mathbb{R}^{n+}$ is the robot maximum joint acceleration vector.
- *Energy limit*: The maximum energy of the robot is limited based on the maximum exchange of energy imposed by the safety regulations, i.e. equation (8), on the entire trajectory $\mathbf{q}(\cdot)$. This allows ensuring that, even when the robot is moving exactly towards the human operator, a possible collision will not harm the human operator.
- *Collision constraint*: During the task execution, collisions between the robot itself and between the robot and the environment need to be avoided.

For this purpose the following collision matrices have been defined:

- $C_r \in \mathbb{N}^{n \times n}$ as the robot self collision matrix, where the generic $c_{ri,j}$ element is 1 if the collision between the i-th link and the j-th link of the robot must be avoided, and 0 if not.
- $C_e \in \mathbb{N}^{n \times N_o}$ as the robot-environment collision matrix, where the generic $c_{ek,m}$ element is 1 if the collision between the k-th link of the robot and the m-th obstacle must be avoided, and 0 if not, with $N_o \in \mathbb{N}$ representing the number of obstacles in the environment.
- $C \in \mathbb{N}^{n \times (n+N_o)}$ as the task collision matrix, defined as

$$C = [C_r, C_e] \tag{14}$$

Then, once the distance between links and between a link and an obstacle is available, collisions can be avoided by setting.

$$\begin{array}{ll} d_i^j \geq d_{sft} & \forall (i,j) : c_{ri,j} = 1 \\ d_k^m \geq d_{sft} & \forall (k,m) : c_{ek,m} = 1 \end{array} \tag{15}$$

where $d_i^j \in \mathbb{R}^+$ is the distance between the i-th link o and the j-th link of the robot, $d_k^m \in \mathbb{R}^+$ is the distance between the k-th link of the robot and the m-th obstacle in the environment, and $d_{sft} \in \mathbb{R}^+$ is a user-defined positive parameter representing the safety distance. The matrices C_r and C_e are upper triangular matrices with zero diagonal and are introduced to avoid collision checking between parts of the system that cannot physically collide, avoiding unnecessarily burdening the optimization problem.

C. Cost function

As specified in Section I, the purpose of the optimizer is to find the trajectory that can be traveled in the shortest time. For this reason, the cost function is defined as:

$$J(\mathcal{M}) = T^2 \tag{16}$$

D. Optimization problem

Finally, the solution of the following optimization problem:

$$\begin{aligned} \min_{\mathcal{M}} \; J(\mathcal{M}) \;\; s.t. \;\; & -\dot{\mathbf{q}}_{max} \leq \dot{\mathbf{q}}(\cdot) \leq \dot{\mathbf{q}}_{max} \\ & -\ddot{\mathbf{q}}_{max} \leq \ddot{\mathbf{q}}(\cdot) \leq \ddot{\mathbf{q}}_{max} \\ & \Delta \mathbf{K}_e \leq E_{max} \\ & d_i^j \geq d_{sft} \quad \forall (i,j) : c_{ri,j} = 1 \\ & d_k^m \geq d_{sft} \quad \forall (k,m) : c_{ek,m} = 1 \\ & a \geq \frac{4 L_{\mathcal{P}}}{T^2} \end{aligned} \tag{17}$$

returns the optimal parameter set $\mathcal{M}$ and, therefore, the optimal trajectory $\mathbf{q}(\cdot)$.

As can be seen from equation (17), the optimum problem to be solved is strongly non-linear. This is mainly due to the energy constraint (8), which requires the calculation of quantities such as the inertia matrix and the Jacobian that vary non-linearly along the trajectory, and the distance constraint (15) for the collision avoidance, which also varies non-linearly along the iterated trajectory. Solving problems like this requires the use of special solvers, which typically require a lot of computation time and a good and feasible initial guess. The latter, in particular, is of paramount importance as it drastically affects the quality of the solution. Therefore, the solution of the problem in a single stage is not effective. For this reason, it was decided to split the solution of the problem into different stages, as reported in the next Section.

V. Planning strategy

Given the high non-linearity of the optimization problem, the planning strategy is composed of three stages.

A. Searching for an initial guess

When solving non-linear optimization problems, the quality of the solution typically depends on the initial point at which the optimizer is started. The goal of this first stage is to find a good candidate for the initial value of $\mathbf{q}_{ref}$, i.e. $\bar{\mathbf{q}}_{ref}$, with which to start solving the optimization problem (17). The initial guess search is implemented according to the pseudo-code reported in Alg. 1. The algorithm starts by setting setting:

$$\mathbf{q}^{\star}_{ref} = \frac{\mathbf{q}_i + \mathbf{q}_f}{2} \tag{18}$$

which represents the intermediate configuration between the start configuration $\mathbf{q}_i$ and the goal configuration $\mathbf{q}_f$, i.e. the value of $\mathbf{q}_{ref}$ which would produce a straight line trajectory in the joint space.

The idea is to generate a mesh grid of configurations close to $\mathbf{q}^{\star}_{ref}$, with the desired size of the mesh $S_a \in \mathbb{R}$, and choose the best one. This strategy is applied $N_f \in \mathbb{N}$ times starting from the last best value of $\mathbf{q}^{\star}_{ref}$, and decreasing the spacing of the grid of $S_a^{\%} \in \mathbb{R}$ at each iteration. To find the best configuration (Line 8), it has been chosen to evaluate a given configuration as the trace of the inertia matrix that the robot exhibits in that configuration, namely:

$$\psi = tr(M(\mathbf{q})) \tag{19}$$

with $\psi \in \mathbb{R}$, and choosing the configuration that produces a lowest value of ψ. This choice is based on the principle that the trace is directly related to the eigenvalues of the inertia matrix, and that, for the same velocity, the bigger

Algorithm 1 qRefSearch

1: **Require:** $\mathbf{q}_i, \mathbf{q}_f, S_a^0, N_f, N_l, S_a^{\%}$
2: $\mathbf{q}^{\star}_{ref} \leftarrow (\mathbf{q}_i + \mathbf{q}_f)/2$
3: $S_a \leftarrow S_a^0$
4: **for** $k \leftarrow 1$ to N_f **do**
5: $\quad \mathbf{q}_{lb}, \mathbf{q}_{ub} \leftarrow \mathbf{selectSearchSpace}(\mathbf{q}^{\star}_{ref}, S_a)$
6: $\quad \mathcal{T} \leftarrow \mathbf{generateGrid}(\mathbf{q}_{lb}, \mathbf{q}_{ub}, N_l)$
7: $\quad \mathcal{T}_{\mathcal{F}} \leftarrow \mathbf{findFeasible}(\mathcal{T})$
8: $\quad \mathbf{q}^{\star}_{ref} \leftarrow \mathbf{findBest}(\mathcal{T}_{\mathcal{F}})$
9: $\quad S_a \leftarrow S_a S_a^{\%}$
10: **end for**
11: $\bar{\mathbf{q}}_{ref} \leftarrow \mathbf{q}^{\star}_{ref}$

the eigenvalues the bigger is the kinetic energy. Therefore, decreasing the trace is equivalent to decreasing the inertia, which allows the trajectory to be traversed with greater velocity, decreasing the execution time. This is the direction that most likely the optimizer will look for the optimal solution.

The initial value of S_a is initialized to $S_a^0 \in \mathbb{R}$ and the searching loop starts. First, the grid bounds $\mathbf{q}_{lb}$ and $\mathbf{q}_{ub}$ are calculated as a function of S_a and the reference configuration $\mathbf{q}^\star_{ref}$. After that, the search grid $\mathcal{T}$ is generated by splitting the search limits into $N_l \in \mathbb{N}$ levels.

The feasible configurations $\mathcal{T}_\mathcal{F}$ are selected from the grid $\mathcal{T}$, i.e. the configurations such that the paths of the corresponding trajectories are collision-free.

If the grid bounds $\mathbf{q}_{ub}$ and $\mathbf{q}_{lb}$ are such that they include the joint limits of the robot and the algorithm returns an empty subset $\mathcal{T}_\mathcal{F}$, it can be that a) an intermediate configuration $\mathbf{q}^\star_{ref}$ does not exist such that the spline $\mathcal{P}(s)$ starting from $\mathbf{q}_i$, ending in $\mathbf{q}_f$ and passing through $\mathbf{q}^\star_{ref}$ is collision free or b) the spacing of the grid S_a is not sufficiently fine.

If the subset $\mathcal{T}_\mathcal{F}$ is not empty, the best $\mathbf{q}^\star_{ref}$ is selected by computing (19) for each configuration in $\mathcal{T}_\mathcal{F}$ and choosing the one with the lower value of ψ.

At the end of each iteration, the search is decreased with S_a of $S_a^\%$, in order to focus the search near $\mathbf{q}^\star_{ref}$.

Finally, the value of the candidate initial guess $\bar{\mathbf{q}}_{ref}$ is returned.

B. Pre-optimizing

As mentioned in the previous Section, the initial guess is searched in terms of $\mathbf{q}_{ref}$ only, neglecting the parameters of the velocity profile a and T. In order to provide an initial guess to the optimization problem (17), it is necessary to find a good initial guess also for these two quantities as well. To obtain them, the trajectory $\mathbf{q}(\cdot)$ is forced to pass through $\bar{\mathbf{q}}_{ref}$, and the following optimization problem is solved:

$$\begin{aligned} \min_{\bar{a},\,\bar{T}} \; T^2 \qquad s.t. \;\; & -\dot{\mathbf{q}}_{max} \leq \dot{\mathbf{q}}(\cdot) \leq \dot{\mathbf{q}}_{max} \\ & -\ddot{\mathbf{q}}_{max} \leq \ddot{\mathbf{q}}(\cdot) \leq \ddot{\mathbf{q}}_{max} \\ & \Delta\mathbf{K}_e \leq E_{max} \\ & \bar{a} \geq \frac{4L_\mathcal{P}}{\bar{T}^2} \end{aligned} \tag{20}$$

namely, it computes the values of $\bar{a}$ and $\bar{T}$ such that the time is minimized and the feasibility of velocities and acceleration are guaranteed under the energy limit.

C. Optimizing

The real optimization is performed in this stage by solving the problem given in (17), settings as initial guess the set of parameters $\bar{\mathcal{M}} = \{\bar{\mathbf{q}}_{ref}, \bar{a}, \bar{T}\}$ obtained from the previous two stages.

VI. Validation

A. Simulations

To test the effectiveness of the proposed planner, we selected 250 combinations of initial configurations and final configurations, and for each pair, we planned the corresponding trajectory.

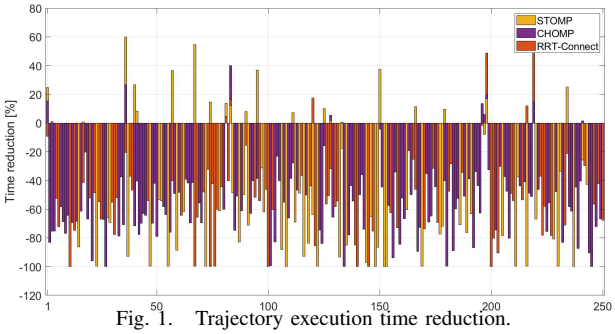

Fig. 1. Trajectory execution time reduction.

For this validation, we chose to use the Universal Robot's UR10e 6-DoFs collaborative robot and to consider only the robot's self-collision as the possible collision during the movement.

The 250 combinations were randomly generated with joint values between $\pm 2\pi$, ensuring that for all generated configurations the robot was not in self-collision. In addition, all trajectories were avoided for which, a priori, it was not possible to move the robot without generating a collision. In the specific case of the UR10e universal robot, those with joint 2 transited at π.

We consider a scenario for which the risk assessment study provides the following considerations:

- collisions with the human can only occur on the back and shoulder, for which the ISO/TS 15066 provides a maximum transferred energy $\Delta\mathbf{K}_e = 2.5J$ and a mass for the human of $m_H = 40.0Kg$
- the human has a maximum speed in the robot's direction of $\dot{\mathbf{x}}_{H_n} = 0.5m/s$

To compare the performance of the proposed planner with the state of the art, we apply the following strategy:

1) we selected a set of state-of-the-art motion planning algorithms, namely RRT-Connect, CHOMP, and STOMP. We will refer hereafter to this set as *soa planners*.
2) for each initial and final configuration, we planned a path using the *soa planners*
3) to obtain a faithful comparison, each path has been parametrized in time by selecting a trapezoidal speed profile that is compliant with the safety constraint, i.e. by means of (20).

The proposed planner was developed using the MATLAB®environment with the global optimization and robotics toolboxes. We chose to use the patternsearch algorithm [22] in the pre-optimizer and optimizer.

Fig. 1 reports the time reduction percentage achieved with the proposed planner compared to the *soa planners*, from which we measure an average of -43.93% compared to RRT-Connect, an average of -49.58% compared to CHOMP, and an average of -39.90% compared to STOMP.

It is worth noting that the calculation time of a single trajectory with the proposed planner is around 289s while the calculation time for the *soa planners* is negligible. Nevertheless, the purpose of the proposed planner is to provide

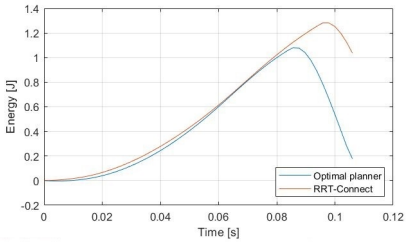

Fig. 2. Energy dissipated during the impact

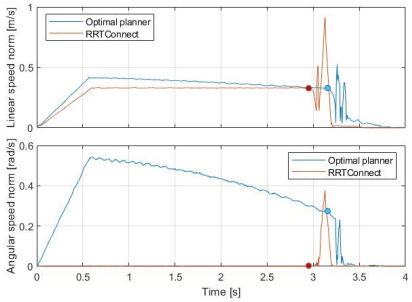

Fig. 3. Linear and angular speed norm during the experiment.

optimal trajectories for industrial tasks whose trajectories are computed during the cell design phase. For this reason, the calculation time obtained is considered acceptable. Furthermore, the optimization time can be reduced by exploiting other programming languages instead of Matlab, e.g. C++.

B. Experiments

To test the effectiveness of the proposed planner also in practical cases, the following experiment was chosen: 1) the initial configuration was set equal to $\mathbf{q}_i = [0, -\pi/2, 0, -\pi/2, \pi/2, 0]^T$, 2) the final configuration was set equal to $\mathbf{q}_f = [0, -3\pi/2, 0, -\pi/2, \pi/2, 0]^T$, 3) we plan two different path: the first one using the proposed planner, and the second one using the RRT-Connect planner, which returns the same path of STOMP and CHOMP for this particular case, and 4) each trajectory was performed in such a way that a collision occurred between the end-effector and a fixed object. Using a force/torque sensor at the end-effector of the robot, we measured the energy dissipated during the collision. The fixed object was placed differently according to the planned trajectory so that the impact occurred at the same speed for both trajectories. A UR10e universal robot was also chosen for the experimental validation. Moreover, the configuration of the safety parameters was not changed compared to the simulations. Finally, the choice of impacting a fixed object instead of a physical person was made in order to minimize the impact difference between one case and the other, which is affected by the different human reactions in the two cases. Fig. 2 shows the results obtained in terms of energy dissipated during the impact, while Fig. 3 shows the linear and angular velocity norm throughout the trajectory. In Fig. 3, the blue and red markers indicate the instants at which the impact starts, for the trajectory planned with the proposed planner and the RRTConnect planner, respectively. As can be seen in Fig. 3, both impacts occur at

Fig. 4. Impact configurations. On the left, using the propose planner. On the right, using the RRT-Connect planner.

approximately the same linear velocity of $0.32m/s$, and with a higher angular velocity with the trajectory planned using the proposed planner ($0.27rad/s$ vs. $0.0rad/s$). Despite this, looking at Fig. 2, the energy dissipated in the impact with the RRT-Connect trajectory is greater than the one dissipated with the trajectory planned with the proposed planner. This is because the trajectory planned with the proposed planner travels through configurations with a lower inertia and higher velocity, allowing the trajectory to be traveled in less time but with the same energy limit.This can be seen in Fig. 4, which shows the two configurations during impact. As can be seen, with the proposed planner the robot impacts with a more crouched configuration (which translates into a lower inertia matrix), whereas with the RRT-Connect planner, it impacts with its arm fully extended (which translates into a higher inertia matrix). The choice of having the robot impact at the same speed was dictated by the desire to minimize the bias introduced by the activation of the robot's safety functions. In fact, for same activation time, a higher impact speed impact result in a longer interaction time, i.e. more energy dissipated. This result allows us to state that the proposed planner returns trajectories with less inertia and that, at the same velocity, transmits less energy during impact. Therefore, it can be said that for the same amount of energy transmitted, the proposed planner allows the trajectory to go faster, which allows the total time of the trajectory to decrease.

VII. Conclusions and Future Works

This paper presents a new time-optimal planner for safe human-robot collaboration. The algorithm considers the PFL constraint from ISO/TS 15066, defining the maximum admissible energy for robot-human collaboration. This limit is then integrated into a three-stage optimization problem to determine the optimal trajectory in terms of execution time. The algorithm is compared with RRT-Connect, STOMP, and CHOMP in simulation and real scenarios, demonstrating superior performance with lower execution time and energy exchange.

Future works will target improving the computational complexity, which is the main drawback of the approach. Addressing this issue will allow analyzing how the system behaves when changing the parameters of the optimizer, e.g. the order of the spline or the velocity profile. Furthermore, the way of choosing the initial guess can be modified and integrated with another planner, such as RRT-Connect. Lastly, the planner strategy may be compared with other soa algorithms [23] and tested in more complex scenarios.

REFERENCES

[1] V. Villani, F. Pini, F. Leali, and C. Secchi, "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications," *Mechatronics*, vol. 55, pp. 248–266, 2018.

[2] *ISO 10218-1:2011(E). Robots and Robotic Devices–Safety Requirements for Industrial Robots–Part 1: Robots*, International Organization for Standardization Std., Jul. 2011.

[3] *ISO 10218-2:2011(E). Robots and Robotic Devices–Safety Requirements for Industrial Robots–Part 2: Robot systems and integration*, International Organization for Standardization Std., Jul. 2011.

[4] *ISO/TS 15066:2016(E). Robots and robotic devices–Collaborative robots*, International Organization for Standardization Technical Specification, Feb. 2016.

[5] E. Prianto, M. Kim, J.-H. Park, J.-H. Bae, and J.-S. Kim, "Path planning for multi-arm manipulators using deep reinforcement learning: Soft actor–critic with hindsight experience replay," *Sensors*, vol. 20, no. 20, p. 5911, 2020.

[6] X. Li, H. Liu, and M. Dong, "A general framework of motion planning for redundant robot manipulator based on deep reinforcement learning," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5253–5263, 2021.

[7] L. Jiang, S. Liu, Y. Cui, and H. Jiang, "Path planning for robotic manipulator in complex multi-obstacle environment based on improved_rrt," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 6, pp. 4774–4785, 2022.

[8] R. Rossi, M. P. Polverini, A. M. Zanchettin, and P. Rocco, "A pre-collision control strategy for human-robot interaction based on dissipated energy in potential inelastic impacts," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 26–31.

[9] A. Palleschi, M. Hamad, S. Abdolshah, M. Garabini, S. Haddadin, and L. Pallottino, "Fast and safe trajectory planning: Solving the cobot performance/safety trade-off in human-robot shared environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5445–5452, 2021.

[10] K. Merckaert, B. Convens, C.-j. Wu, A. Roncone, M. M. Nicotra, and B. Vanderborght, "Real-time motion control of robotic manipulators for safe human–robot coexistence," *Robotics and Computer-Integrated Manufacturing*, vol. 73, p. 102223, 2022.

[11] F. Benzi, F. Ferraguti, and C. Secchi, "Energy tank-based control framework for satisfying the iso/ts 15066 constraint," *arXiv preprint arXiv:2304.14059*, 2023.

[12] A. Pupa, M. Arrfou, G. Andreoni, and C. Secchi, "A safety-aware kinodynamic architecture for human-robot collaboration," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4465–4471, 2021.

[13] N. Lucci, B. Lacevic, A. M. Zanchettin, and P. Rocco, "Combining speed and separation monitoring with power and force limiting for safe collaborative robotics applications," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6121–6128, 2020.

[14] F. Ferraguti, M. Bertuletti, C. T. Landi, M. Bonfè, C. Fantuzzi, and C. Secchi, "A control barrier function approach for maximizing performance while fulfilling to iso/ts 15066 regulations," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5921–5928, 2020.

[15] T. Kunz and M. Stilman, "Time-optimal trajectory generation for path following with bounded acceleration and velocity," *Robotics: Science and Systems VIII*, pp. 1–8, 2012.

[16] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[17] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[18] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[19] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[20] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

[21] J. Y. Luh, M. W. Walker, and R. P. Paul, "On-line computational scheme for mechanical manipulators," 1980.

[22] C. Audet and J. E. Dennis Jr, "Analysis of generalized pattern searches," *SIAM Journal on optimization*, vol. 13, no. 3, pp. 889–903, 2002.

[23] T. Kunz and M. Stilman, "Probabilistically complete kinodynamic planning for robot manipulators with acceleration limits," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3713–3719.