2023 I-RIM Conference
October 20-22, Rome, Italy
ISBN: 978-88-945805-4-9
10.5281/zenodo.10722456

Throwing objects with collaborative manipulators using minimum-jerk trajectories

Giorgio Simonini[1,2], Riccardo Di Majo[1], Lorenzo Boccalini[1], Matteo Guerci[1], Antonio Bicchi[1,3], Paolo Salaris[1]

***Abstract*—Recently, throwing is gaining even more research interest, especially in logistics, where goods have to be moved efficiently. The throwing approach is difficult because it requires precision in a dynamic task. In this work, we consider the problem in both joint and cartesian space. In joint space, we use an optimization procedure to obtain a trajectory to follow, maximizing the throwing distance. In task space, we use a cartesian controller to throw in a desired position. In both cases, minimum-jerk trajectories are used to have smooth signals and deal with robot constraints, usually present in collaborative robots. The methods are tested on simulations and experiments on the 7 d.o.f *Franka Emika Panda*.**

I. Introduction

In logistics, throwing objects has been found to be a more effective technique in several situations. For example, it could be more efficient than the pick-and-place action because the robot does not stop for the placing phase. The European project Darko wants to investigate and put into practice this idea by employing a mobile platform to streamline the movement and distribution of commodities. Human being uses throwing many times in their life, i.e. throwing objects into the recycle bin, or moving them quickly from one location to another. As usual, it could give inspiration; authors in [1],[2] use the kinetic chain approach to generate the throwing motion. In general, robotic movement can be designed in cartesian or joint space. The former is more intuitive but not relies on the structure of the robot. The latter is difficult to study but permits usually more robust results. In [3] a nonprehensile throwing on a 2-D surface is developed. In [4] the throwing is used in a recycling factory with a cartesian approach. Authors in [5] solve a nonlinear optimization problem to generate the joint-level motion. Recently, tossingBot in [6] obtain visibility thanks to a learning-based architecture that performs throws with different objects.

In this work, we investigate the throwing problem with the purpose of maximum distance by a joint-level approach, and with precise throwing in a cartesian space. Sec. III define the two throwing problems. In Sec. IV-A the problem is solved in joint-space, using an optimization. In Sec. IV-B the solution is obtained in task space using a cartesian controller. The work is validated on Sec. V on a 7 d.o.f. Panda manipulator. Conclusion on Sec. VI give some result summary and possible future works.

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Grant Agreements No. 101017274 (Darko) and in part by the Ministry of University and Research (MUR) in the framework of the FoReLab and CrossLab projects (Department of Excellence).

[1]Dipartimento di Ingegneria dell'Informazione e Centro di Ricerca "Enrico Piaggio", Università di Pisa, Largo Lucio Lazzarino 1, 56126 Pisa, Italy

[2]Italian Doctorate in Robotics and Intelligent Machines

[3]Soft Robotics for Human Cooperation and Rehabilitation, Fondazione Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy

giorgio.simonini@phd.unipi.it

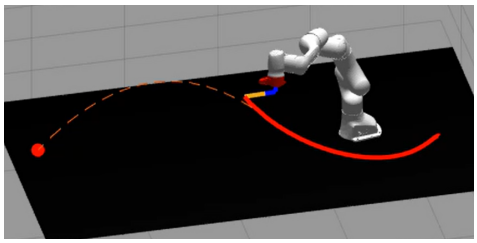

Fig. 1. Throwing task, from an initial position to a desired point, or maximum distance

II. Minimum-jerk trajectory

The minimum-jerk trajectory from the time $t = 0$ to the time t_r on a variable s is created from the minimization of the cost function

$$\int_0^{t_\mathrm{r}} \| \dddot{s}(t) \| dt. \tag{1}$$

That leads to the constraint $s^{(6)} = 0$. The final motion is a 5^{th}-order spline created imposing the initial and final conditions.

III. Problem definiton

In this work, we take two types of problems: a maximum distance and a precise throw. In both cases, the projectile motion without air friction is considered as flying model. Let $p(t) = (x(t), y(t), z(t)) \in \mathcal{R}^3$ be the object center and ξ the end-effector position. Consider that $\xi(t) = p(t)\ \forall t < t_\mathrm{r}$. Robot starts in configuration q_0 at the time $t = 0$, release the object in q_r at time t_r and stops in q_f at time t_f. The point reached by the object is obtained from the flying kinematics, i.e.

$$\begin{cases} x(t) &= x(t_\mathrm{r}) + \dot{x}(t_\mathrm{r})(t_\mathrm{f} - t_\mathrm{r}), \\ y(t) &= y(t_\mathrm{r}) + \dot{y}(t_\mathrm{r})(t_\mathrm{f} - t_\mathrm{r}), \\ z(t) &= z(t_\mathrm{r}) + \dot{z}(t_\mathrm{r})(t_\mathrm{f} - t_\mathrm{r}) - g(t_\mathrm{f} - t_\mathrm{r})^2, \end{cases} \tag{2}$$

where $g \in \mathcal{R}$ is the gravity acceleration. The final time t_f is the time in which the object touches the desired height, and it is

$$t_\mathrm{f} = t_\mathrm{r} + \frac{1}{g}\left(\dot{z}_\mathrm{r} + \sqrt{\dot{z}_\mathrm{r}{}^2 + 2g(z_\mathrm{r} - z_\mathrm{d})}\right). \tag{3}$$

Collaborative robots like *Franka* have often limits in velocity, accelerations and also jerks. The use of minimum-jerk movements permits to have smooth behavior and to deal with such constraints. The trajectory is composed of two

phases: acceleration and brake. The acceleration brings the end-effector to the desired speed. The brake stops the robot's motion. The two problems are described in Sec. III-A and III-B.

A. Maximum distance throwing

The request in this problem is to reach the maximum distance among the x axis by some control input u. The problem is formulated as

$$\max_{u} \; x(t_\mathrm{f}), \tag{4}$$

respecting the manipulator limits, i.e.

$$\begin{aligned} q_\mathrm{MIN} < q < q_\mathrm{MAX}, \qquad & \dot{\xi}_\mathrm{MIN} < \dot{\xi} < \dot{\xi}_\mathrm{MAX}, \\ \dot{q}_\mathrm{MIN} < \dot{q} < \dot{q}_\mathrm{MAX}, \qquad & \tau_\mathrm{MIN} < \tau < \tau_\mathrm{MAX}, \\ \ddot{q}_\mathrm{MIN} < \ddot{q} < \ddot{q}_\mathrm{MAX}, \qquad & \dot{\tau}_\mathrm{MIN} < \dot{\tau} < \dot{\tau}_\mathrm{MAX}. \end{aligned} \tag{5}$$

B. Precise position throwing

In this case, the object has to reach the desired position $p_\mathrm{d} = (x_\mathrm{d}, y_\mathrm{d}, z_\mathrm{d})$, i.e.

$$x(t_\mathrm{f}) = x_\mathrm{d}, \quad y(t_\mathrm{f}) = y_\mathrm{d}, \quad z(t_\mathrm{f}) = z_\mathrm{d} \tag{6}$$

There is no direct need for optimization since the end-effector velocity and position influence directly the throw. Indeed, the range equation is

$$d = \frac{v\cos(\theta)}{g}\Big(v\sin(\theta) + \sqrt{v^2\sin^2(\theta) + 2gh}\Big) \tag{7}$$

where d is the distance, v is the speed, θ is the throwing angle and h is the height difference.

IV. Problem solution

The main point of the solution is to create the desired motion with minimum-jerk trajectories. In the two problems, a low-level controller is developed and the trajectory is fed to it. The controller in both cases is a computed torque in the joints and cartesian space respectively.

A. Joint-space motion

The minimum-jerk trajectory is defined by initial and final points (q_0, q_f), velocities $(\dot{q}_0, \dot{q}_\mathrm{f})$, accelerations $(\ddot{q}_0, \ddot{q}_\mathrm{f})$ and time of execution t_r. In the nonlinear optimization problem (4), free variables are final joint position, velocity and time. For this reason, $u = (q_\mathrm{f}, \dot{q}_\mathrm{f}, t_\mathrm{r})$. The small set of decision variables makes optimization fast. Initial-guess is chosen from an initial sampling of the configuration space. The dataset of configurations $(q_\mathrm{r}, \dot{q}_\mathrm{r})$ is created offline and for each one a throwing position is associated. Then, starting from the farthest distance, the trajectory is created and the first one that does not violate robot constraints is kept as the initial guess. The object distance is obtained from each configuration by forward kinematics $\dot{\xi} = J\dot{q}$ and law (2). Then the optimization procedure refines the solution to find the best release configuration in order to throw. The manipulator brake phase is generated as a constant deceleration and the final end-effector position is considered in the optimization as a constraint.

B. Cartesian-space motion

In this case, the robot is moved by a cartesian trajectory. First, the vertical plane containing the initial point p_0 and the desired point p_d is created. Then, a sphere in the task space, centered on the robot, is defined as a *safe* region to throw. Throwing inside that sphere has no sense because the object could directly be dropped. The intersection between the plane and the sphere provides possible throwing points. The point that requires less velocity, using the optimal throwing angle, is chosen for the throw. The optimal throwing angle depends on the height difference between starting and desired points and it is obtained numerically from the range equation (7). The trajectory from the initial to the release point is a minimum-jerk with the determined final configuration and a reasonable time t_r. The end-effector orientation is chosen with the palm aligned to the velocity vector, and it is interpolated using a *SLEPR*. The final cartesian trajectory is provided to the controller.

V. Validation

We validate the method on simulations and experiments on a 7 d.o.f. robot *Franka Emika Panda*. The use of minimum-jerk trajectories leads to smooth behavior and it permits to not violate limits on jerk and torque derivative on the robot. In the maximum distance task, the robot reaches a distance of 1.67m in simulation and 1.5m in the real robot. In the precise throwing, the robot has an error of $10cm$ on about twenty trials. Experiment and simulation material is available online [1].

VI. Future work

One of the difficult points is to coordinate the gripper to open at the right time and it causes the majority of the error. In this regard, a study in this direction could be made with the use of AI and learning methods.

References

[1] T. Senoo, A. Namiki, and M. Ishikawa, "High-speed throwing motion based on kinetic chain approach," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, ISSN: 2153-0866, Sep. 2008, pp. 3206–3211.

[2] S. Ichinose, S. Katsumata, S. Nakaura, and M. Sampei, "Throwing motion control experiment utilizing 2-link arm passive joint," in *2008 SICE Annual Conference*, Aug. 2008, pp. 3256–3261.

[3] A. Pekarovskiy and M. Buss, "Optimal control goal manifolds for planar nonprehensile throwing," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, ISSN: 2153-0866, Nov. 2013, pp. 4518–4524.

[4] H. Chen, B. Zhang, and T. Fuhlbrigge, "Robot throwing trajectory planning for solid waste handling," in *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, ISSN: 2379-7711, Jul. 2019, pp. 1372–1375.

[5] F. Lombai and G. Szederkenyi, "Throwing motion generation using nonlinear optimization on a 6-degree-of-freedom robot manipulator," in *2009 IEEE International Conference on Mechatronics*, Apr. 2009, pp. 1–6.

[6] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, *TossingBot: Learning to throw arbitrary objects with residual physics*, Issue: arXiv:1903.11239, May 30, 2020. arXiv: 1903.11239[cs,stat].

[1] https://drive.google.com/drive/folders/1A3Arts_rZQLW3ssIC9Vrpc5Vcva6ZI1J?usp=sharing