UNIVERSITAT POLITÈCNICA DE CATALUNYA

Doctoral Programme:

AUTOMÀTICA, ROBÒTICA I VISIÓ

Research Plan:

Kinodynamic Planning for Constrained Robotic Systems

Ricard Bordalba

Advisors:

Lluís Ros, Dr.

Josep M Porta, Dr.

April, 2017

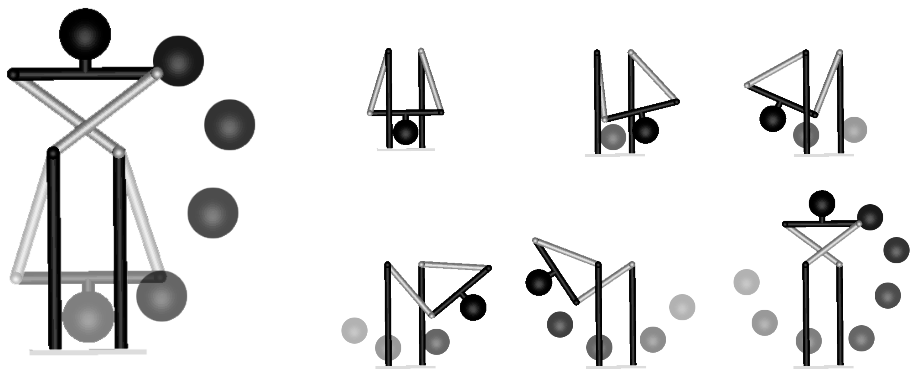
Contents

1 Introduction

The motion planning problem has been a subject of active research since the early days of Robotics [59]. Although it can be stated in simple terms—find a feasible trajectory to move a robot from a start to a goal location—and despite the significant advances in the field, it is still an open problem in many respects. The complexity of the problem arises from the many kinematic and dynamic constraints that have to be taken into account, such as potential collisions with static or moving objects in the environment, loop-closure constraints, velocity constraints, singularity avoidance, torque and velocity limits, or energy and time execution bounds, to name a few. All these constraints are relevant in the factory and home environments in which Robotics is called to play a fundamental role in the near future.

The complexity of the problem is typically tackled by first relaxing some of the constraints. For example, while obstacle avoidance is a fundamental issue, the lazy approaches initially disregard it [13]. Other approaches concentrate on geometric [54] and kinematic feasibility [34], which constitute already challenging issues by themselves. In these approaches, dynamic constraints such as speed, acceleration, or torque limits are neglected, with the hope that they will be enforced in a postprocessing stage [29]. Decoupled approaches, however, may not lead to solutions satisfying all the constraints. It is not difficult to find situations in which a kinematically-feasible trajectory becomes unusable because it does not account for the system dynamics (Fig. 1).

Due to their simplicity, decoupled approaches have been predominant so far. Since the late 90s, however, the steady increment in computing power has made it easier to simultaneously satisfy more and more constraints in the devised planners. With this aim in mind, Donald et al. [28] defined the term *kinodynamic planning* to refer to the general problem of finding a motion that satisfies all kinematic and dynamic constraints, and proposed an initial algorithm for this problem. Their planner only applied to a point robot subject to velocity and acceleration bounds, but triggered the quest towards successively more general methods. In subsequent years, a variety of planners were developed that considered obstacle avoidance, non-holonomic constraints, robot dynamics, and force, velocity and acceleration limits in an increasingly integrated manner.



(a) (b)

Figure 1: A four-bar pendulum modeling a swing boat ride. A kinematic trajectory (a) and a trajectory also fulfilling dynamic constraints due to torque limitation (b) may be quite different. An animation of the trajectory in (b) can be seen in `https://goo.gl/S9UKdn`.

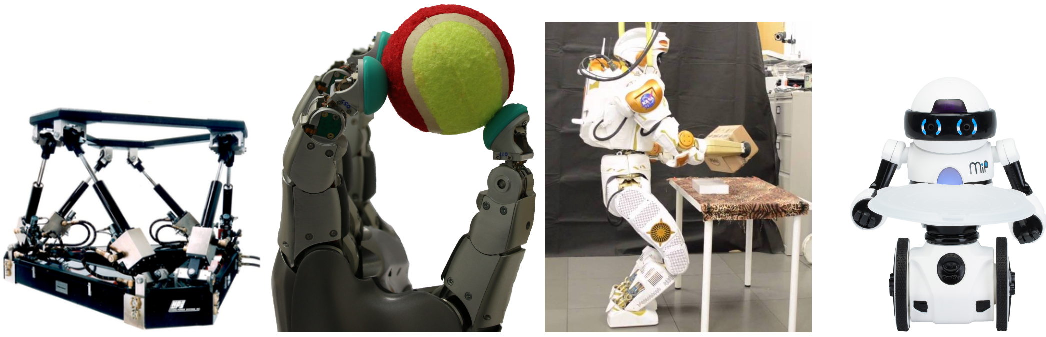


Figure 2: From left to right: a parallel robot, a robot hand manipulating an object, a humanoid picking a box, and a service robot that has to keep a tray horizontally.

As opposed to path planners, which only compute a path in configuration space, kinodynamic planners obtain a time-parametric trajectory in the state space—the set of all position-velocity pairs of the robot. This fact complicates the planning of motions substantially, as it doubles the dimension of the search space in which the planning has to be solved. Even so, existing kinodynamic planners are quite generic and scale well with the problem dimension. The vast majority of such planners, however, have an important limitation: they assume that the state space is globally parametrizable, i.e., that each state can be represented by means of independent generalized coordinates. While parametric state spaces often arise in robotics, for example in robots with tree topology moving in free space, in practice, kinematic constraints may appear that relate the state space coordinates. This happens in many robotic devices, like parallel manipulators, robots in contact with objects or with the environment, or when virtual geometric constraints are needed to fulfill a specific task (Fig. 2). In these cases, the robotic system is said to be *constrained*, as its state space becomes a nonlinear manifold implicitly-defined by the kinematic equations to be satisfied. As we shall see, this fact hinders the application of most motion planning approaches because, then, the state space is not globally parametrizable, simulation of the robot requires the use of differential algebraic equations, instead of ordinary differential equations, and singularities may arise that complicate such simulation. It is probably for these reasons that a mature kinodynamic planner for constrained systems has not been developed to date. The purpose of this PhD. work is, precisely, to help filling this gap to the largest possible extent.

2 Objectives

The goal of this thesis is to provide reliable algorithms for solving the kinodynamic motion planning problem on constrained robotic systems. Given a kinematic and dynamic model of the system, and a geometric model of the environment, this problem consists in computing the required force inputs needed to move a robot between two prescribed states, while respecting the following constraints:

- **Kinematic constraints** which are those only involving position and velocity coordinates of the mechanism. These include constraints due to sliding and rolling contacts between bodies, closed kinematic loops inherent to the robot structure or to the task to be executed, collision avoidance, and joint limits.

- **Dynamic constraints** which are those involving position, velocity and acceleration coordinates, and the forces acting on the system. These correspond to the equations of motion of the robot, limits on the actuator or constraint forces, or existing velocity and acceleration bounds.

Although solutions to the kinodynamic planning problem have been given in the past, to the best of our knowledge, no satisfactory approach has been given for constrained systems whose state space is not globally parametrizable. Therefore, the kinodynamic planning problem remains open for the class of systems herein considered.

3 Scope and Assumptions

For the purpose of this work, a robotic system will be a multibody system composed of rigid bodies and lower-pair joints, with some of the joints being actuated. We shall restrict our attention to constrained robotic systems, i.e., those for which their state space is implicitly defined by a system of kinematic constraints involving position and velocity variables. Such constraints can reflect physical joints, contacts, closed kinematic chains, or virtual geometric constraints imposed by the task to be performed. In any case, all of these constraints will be considered to be permanent, as opposed to intermittent constraints that arise, for example, when the robot makes or breaks contact with the ground. Thus, nonsmooth phenomena due to impact dynamics will not play a role in our study.

Our main focus will be on fully-actuated robots, i.e., those with as many actuators as degrees of freedom to be controlled, but the resulting methods should also be applicable to overactuated or underactuated robots in principle. In most cases, the actuator forces will be limited to a prescribed range and limits may also be imposed in internal constraint forces. While the former account for limited actuation capacity, the latter are used to guarantee the materials' resistance, or the proper functioning of the robot.

The entire approach will be model-based. We shall assume that proper models of the robot and its environment are available. This implies that the robot dimensions and dynamic parameters, as well as the geometry and location of all obstacles, will be known with sufficient accuracy. Thus, the problems of system identification and calibration will be out of the scope of our research.

The goal of the planner will be to compute actuator forces able to bring the robotic system from a start to a goal location. The output of the planner will be a time-parametric description of such forces and its resulting state space trajectory, which should respect all kinematic and dynamic constraints imposed by the problem. The development of controllers that might be necessary to finally execute the planned trajectory will also be beyond the research focus of this work.

The planner algorithm should satisfy the following requirements. First, it should find a trajectory whenever one exists, and enough computing time is available. Second, the trajectory should be reasonably smooth to allow a fine control during its execution. Finally, the planner should implement mechanisms to reduce the cost of the trajectory to the largest possible extent, either in terms of time spent or energy consumed by the robot.

4 Expected Contributions

This Ph.D. work seeks to contribute to the general understanding on how the motions of constrained systems can be computed, simulated, or planned in an efficient and reliable way. The main contribution of this research will be a kinodynamic planner for constrained robotic systems. The resulting algorithms will be implemented and integrated in the open source CUIK suite library [78].

The availability of such a planner would allow a robot to automatically convert high-level specifications of a task into low-level descriptions on how to achieve the task. This planner would mainly find applications in the following domains:

- In Robotics, the planner could be used to synthesize a nominal trajectory for constrained systems, such as closed-chain manipulators, non-holonomic robots, or humanoids. This is crucial in tasks that are non-repetitive, where a different trajectory is needed for every scenario, for instance, in medical surgery, search-and-rescue operations, or ocean and space exploration. In these contexts, the planner could be used to find a reference trajectory to maneuver a robot around obstacles. This trajectory could be latter executed and stabilized in real-time by using a feedback policy. Moreover, planner trajectories could be used to evaluate the robot design in simulation in order to make sure that it performs properly in different scenarios. In this way, the time and cost expenses of prototyping could be reduced. For example, the planner could conclude that a grasping device is not powerful enough to move a large object, thereby determining that a better design is needed.
- In the game and movie industries, the obtained trajectories could be used to automate the motion of virtual characters, while providing physical realism. For example, a game developer could program a task at a higher level, and the planner would automatically determine the movement of an animated character in an intelligent way. This also becomes useful in computer graphics, for instance, when hundreds of digital actors in a movie should move in an scenario with obstacles. The planner would avoid the time-consuming task of explicitly defining a motion for each actor.

5 Problem Formalization

The kinodynamic planning problem typically takes place in the state space of the robot, i.e., the set $\mathcal{X}$ of kinematically-valid states $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})$, where $\boldsymbol{q}$ is a vector of n_q generalized coordinates describing the configuration of the robot, and $\dot{\boldsymbol{q}}$ is the time derivative of $\boldsymbol{q}$, which describes its velocity. The coordinates in $\boldsymbol{q}$ may be independent or not. In the former case, any pair $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{2n_q}$ is kinematically valid, and $\mathcal{X}$ becomes parametrically defined. The latter case is more complex. The configuration space (C-space) of the robot is the set $\mathcal{C}$ of points $\boldsymbol{q}$ that satisfy a system of n_e nonlinear equations

$$\boldsymbol{\Phi}(\boldsymbol{q}) = \boldsymbol{0} \tag{1}$$

encoding, e.g., joint assembly, geometric, or contact constraints, either inherent to the robot design or necessary for task execution. The constraints in Eq. (1) are said to be holonomic constraints and only depend on position variables. By differentiating Eq. (1), the valid values of $\dot{\boldsymbol{q}}$ are those that fulfill

$$\boldsymbol{\Phi}_{\boldsymbol{q}}(\boldsymbol{q})\, \dot{\boldsymbol{q}} = \boldsymbol{0}, \tag{2}$$

where $\boldsymbol{\Phi}_{\boldsymbol{q}} = \partial\boldsymbol{\Phi}/\partial\boldsymbol{q}$. Likewise, the robot may also be subject to a system of n_h nonholonomic constraints

$$\boldsymbol{A}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{0}, \tag{3}$$

which are velocity constraints that cannot be integrated, i.e., they cannot be expressed as a position constraint like Eq. (1). The consequence of this property is that a nonholonomic robot has more degrees of freedom in position than in velocity. Eqs. (2) and (3) can be combined to form the velocity constraint

$$\begin{bmatrix} \boldsymbol{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}) \\ \boldsymbol{A}(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} = \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{0}, \tag{4}$$

where $\boldsymbol{B}(\boldsymbol{q})$ is an $(n_e + n_h) \times n_q$ matrix, which, under mild conditions, can be assumed to be full rank.

Let $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ denote the system formed by Eqs. (1) and (4). Then, the state space $\mathcal{X}$ of the constrained system becomes a nonlinear manifold of dimension $d_{\mathcal{X}} = 2(n_q - n_e) - n_h$ defined implicitly as

$$\mathcal{X} = \{\boldsymbol{x} : \boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}\}. \tag{5}$$

Any motion planned in $\mathcal{X}$ must also obey the dynamic equations of the robot, which arise from considering the forces and physical laws that determine the system movement. These equations can be written in the form

$$\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}), \tag{6}$$

where $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ is an appropriate differentiable function, and $\boldsymbol{u}$ is a n_u-vector of actuator forces subject to lie in a bounded subset $\mathcal{U} \subset \mathbb{R}^{n_u}$. For each value of $\boldsymbol{u}$, Eq. (6) defines a vector field over $\mathcal{X}$, which can be used to integrate the robot motion forward in time, using proper numerical methods.

In order to obtain Eq. (6), constrained systems are usually modeled with the multiplier form of the Euler-Lagrange equations [32]. First the systems are treated as unconstrained systems by cutting their kinematic loops, and then the loop closure constraints are enforced by using Lagrange multipliers. The dynamic equations then take the form

$$\frac{d}{dt}\frac{\partial K}{\partial \dot{\boldsymbol{q}}} - \frac{\partial K}{\partial \boldsymbol{q}} + \frac{\partial U}{\partial \boldsymbol{q}} + \boldsymbol{B}^{\mathsf{T}}\boldsymbol{\lambda} = \boldsymbol{\tau}, \tag{7}$$

where $\boldsymbol{\lambda}$ is a vector of $n_e + n_h$ Lagrange multipliers, $\boldsymbol{\tau}$ is the generalized force corresponding to the non-conservative forces applied on the system, and K and U are the expressions of the kinetic and potential energies of the robot.

The kinetic energy of the robot can always be defined compactly as a quadratic function of $\dot{\boldsymbol{q}}$, that is

$$K = \frac{1}{2}\dot{\boldsymbol{q}}^{\mathsf{T}}\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{8}$$

where $\boldsymbol{M}(\boldsymbol{q})$ is the so-called mass matrix, which is always symmetric and positive definite. The potential energy $U = U(\boldsymbol{q})$ is independent of $\dot{\boldsymbol{q}}$. These properties allow Eq. (7) to be written in the form

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{B}^{\mathsf{T}}\boldsymbol{\lambda} = \boldsymbol{\tau}, \tag{9}$$

where $\boldsymbol{G}(\boldsymbol{q})$ is a vector of conservative forces (e.g. gravity or spring forces) given by

$$\boldsymbol{G}(\boldsymbol{q}) = \frac{\partial U}{\partial \boldsymbol{q}}, \tag{10}$$

and $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is a vector corresponding to Coriolis and centrifugal forces, which is given by

$$\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = (\boldsymbol{M}_{\boldsymbol{q}}\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{M}_{\boldsymbol{q}}\dot{\boldsymbol{q}}. \tag{11}$$

Since Eq. (9) is a system of n_q equations in $n_q + (n_e + n_h)$ unknowns (the values of $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\lambda}$), we need additional equations to be able to solve for $\ddot{\boldsymbol{q}}$. These can be obtained by differentiating Eq. (4), which yields

$$\boldsymbol{B}\ddot{\boldsymbol{q}} - \boldsymbol{\xi} = \boldsymbol{0}, \tag{12}$$

where $\boldsymbol{\xi} = -(\boldsymbol{B}_{\boldsymbol{q}}\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$. Eqs. (9) and (12) can then be written as

$$\begin{bmatrix} \boldsymbol{M}(\boldsymbol{q}) & \boldsymbol{B}^\top \\ \boldsymbol{B} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \boldsymbol{\xi} \end{bmatrix}. \tag{13}$$

Clearly, if $\boldsymbol{B}$ is full rank, the matrix on the left-hand side of Eq. (13) is invertible, and thus we can write

$$\ddot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{I}_{n_q} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{M}(\boldsymbol{q}) & \boldsymbol{B}^\top \\ \boldsymbol{B} & \boldsymbol{0} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\tau} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \boldsymbol{\xi} \end{bmatrix}. \tag{14}$$

If $\boldsymbol{B}$ is not full rank at the given $\boldsymbol{x}$, we say that $\boldsymbol{x}$ is a state-space singularity. At such a point, it is impossible to write Eq. (13) into the form of Eq. (14). It is clear, then, that state-space singularities should be avoided if Eq. (13) is to be solved during the planning process. Fortunately, state-space singularities are nongeneric phenomena that can be avoided by judicious mechanical design [12], or through the addition of singularity-avoidance constraints in Eq. (1) [11].

To obtain Eq. (6), we finally transform Eq. (14) into a first-order ordinary differential equation using the change of variables $\dot{\boldsymbol{q}} = \boldsymbol{v}$, which yields

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \dot{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{f}(\boldsymbol{q}, \boldsymbol{v}, \boldsymbol{u}) \end{bmatrix} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}). \tag{15}$$

Since in practice the actuator forces are limited, $\boldsymbol{u}$ is always constrained to take values in some bounded subset $\mathcal{U}$ of $\mathbb{R}^{n_u}$, which restricts the range of possible state velocities $\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ at each $\boldsymbol{x} \in \mathcal{X}$. During its motion, moreover, the robot cannot incur in collisions with itself or with the environment, so that the feasible states $\boldsymbol{x}$ are those lying in a subset $\mathcal{X}_{\text{feas}} \subseteq \mathcal{X}$ of non-collision states, where position, velocities, and constraint forces are within given bounds.

With the previous definitions, the planning problem we confront can be phrased as follows. Given two states of $\mathcal{X}_{\text{feas}}$, $\boldsymbol{x}_s$ and $\boldsymbol{x}_g$, find an action trajectory $\boldsymbol{u} = \boldsymbol{u}(t) \in \mathcal{U}$ such that:

- The system trajectory $\boldsymbol{x} = \boldsymbol{x}(t)$ determined by Eqs. (1), (4) and (6) for $\boldsymbol{x}(0) = \boldsymbol{x}_s$, fulfills $\boldsymbol{x}(t_f) = \boldsymbol{x}_g$ for some time $t_f > 0$, and $\boldsymbol{x}(t) \in \mathcal{X}_{\text{feas}}$ for all $t \in [0, t_f]$.
- $\boldsymbol{x}(t)$ is once-differentiable at least, which implies that the computed trajectory will be smooth in position and velocity, and continuous in acceleration.
- The additive cost of executing the trajectory

  $$C(\boldsymbol{x}(0), \boldsymbol{u}(t)) = \int_0^{t_f} c(\boldsymbol{x}(t), \boldsymbol{u}(t))\, dt \tag{16}$$

  is, at least, locally optimal for some given instantaneous cost function $c(\boldsymbol{x}(t), \boldsymbol{u}(t))$.

Note that the previous problem can be considered as a full motion planning problem, as opposed to a *path planning* problem that only asks for a connecting curve in the C-space, without reference to the dynamics of the robot. Following [28, 63] we shall use the term kinodynamic planning to refer to such a planning problem. Note however that, contrary to [28, 63] we allow the presence of Eqs. (1) and (4) in the problem, which makes it more general and challenging at the same time.

6 State of the Art

This section reviews the main kinodynamic planning approaches developed to date. Along the way, we shall see the strengths and weaknesses of the various methods, justifying the choice of approach in our PhD work (Sec. 6.1). Since motion simulation is a key ingredient of many planners, we shall also devote some attention to survey relevant results in this field (Sec. 6.2).

6.1 Kinodynamic Planning

Existing strategies for kinodynamic planning can be grouped into decoupled approaches, which search for a C-space path and then design a dynamic trajectory along this path; and direct approaches, which search for such a trajectory over the state space.

6.1.1 Decoupled Planning

The bulk of such methods concentrate on solving the path planning problem, which aims to find a collision-free path. For example, algebraic approaches, like those based on silhouettes [19] or cell decompositions [15], are complete methods, i.e., they provide a path if one exists and show failure, otherwise. The former define a roadmap of the C-space, and the latter divide this space into collision-free cells. However, both approaches can only deal with low-dimensional problems. Approximate cell decomposition methods [59] only partially alleviate this issue.

Potential fields have better scalability [55]. They follow an attractive potential towards the goal, while avoiding repulsive potentials from the obstacles. However, they suffer from falling into local minima. This issue is addressed by the randomized potential planner proposed in [2], where random walks are used to escape from such minima. Nonetheless, potential field methods require C-space representations of the obstacles or, at least, a metric to measure the distance from the robot to the obstacles, which are not easy to obtain in general.

Sampling-based methods arise as an alternative, since they only require a method to check whether a sampled configuration is in collision, and not the actual distance to the obstacles. Such methods can cope with high-dimensional problems and are probabilistically complete, i.e., they guarantee to find a feasible solution, if one exists and sufficient computing time is available. The two most popular methods among them are the probabilistic roadmap (PRM) [54] and the rapidly-exploring random tree (RRT) [62] methods. The PRM method takes random samples from the C-space and connects them to form a roadmap. Then, the start and goal configurations are added to this roadmap and a graph search algorithm, such as Dijkstra's [27], is used to find a path between these configurations. The RRT method incrementally grows a tree rooted at the start configuration by generating random samples. Each sample is connected to the nearest configuration in the tree according to a given metric. The selection of this metric plays a fundamental role in the efficiency of the approach. The search is completed when the tree reaches the goal. Another relevant tree-based algorithm, the expansive search tree (EST) [42] is less metric-dependent. For every node, it simply measures the local density of neighboring nodes, and uses this density to grow the tree towards unsampled areas.

In any case, the complexity of the path planning problem increases when the robot includes kinematic constraints in the form of Eq. (1), as the valid configurations define a manifold embedded in a given ambient space. Algebraic approaches can deal with such manifolds, but do not scale properly [15, 19], or are limited to particular robot architectures [85]. Thus, the usual approach to address these problems is to extend the common sampling-based methods. The performance of those methods heavily relies on being able to uniformly sample the manifold to be explored. In some robotic systems, distance-based formulations provide global parametrizations that can be used to uniformly sample the C-space [35, 90]. However, since global parametrizations are not available in general, alternative sampling strategies have been devised. For example, Han and Amato [34] sample a subset of joint variables and use inverse kinematics to find values for the remaining ones. Unfortunately, this strategy is not applicable to all robotic systems and, although some improvements have been proposed [25], the probability of generating invalid samples is significant. Also, the non-uniqueness of the solutions for the inverse kinematic problem and the presence of singularities complicate the approach [31]. Task-space planners [6, 83, 94] are similar in the sense that they sample a subset of the variables (those related with the end-effector), although they typically determine values for the remaining variables using numerical techniques instead of closed kinematic functions. Thus, they share the problems of kinematic-based approaches regarding the multiple solutions for the non-fixed variables. Another strategy is to sample in the ambient space and then try to converge to the C-space [5, 26, 87, 93, 95]. However, a uniform distribution of samples in the ambient space does not translate into a uniform distribution in the C-space [6]. A better alternative is to sample on an approximation of the constraint manifold, either learned from a collection of valid samples [39], inferred from the nodes of an exploration tree [95], or constructed from tangent-space parametrizations [45, 77, 89]. The latter technique provides better approximation and thus, a more uniform sampling and a more efficient exploration of the manifold.

In the aforementioned approaches, the obtained path is fed into a trajectory generator. Some methods gradually modify this path to obey the dynamic constraints by using optimization techniques [58]. Others find a optimal time scaling for the path subject to dynamic constraints [10, 66, 71, 82, 86]. These methods have been successfully applied to solve complex tasks, such as the coordination of mobile robots [69], or the stabilization of legged robots subject to balance constraints [38, 72]. In any case, decoupled approaches may lead to highly suboptimal solutions involving difficult maneuvers, or even worse, to dynamically unfeasible solutions. For instance, the kinematically-feasible path in Fig. 1a becomes unusable because it does not consider actuator torque limits and cannot be executed in practice. Instead, the dynamically-feasible trajectory in Fig. 1b overcomes this limitation by increasing the momentum of the load in order to reach the goal after a few oscillations. To synthesize such trajectories, thus, one needs to resort to direct planning methods.

6.1.2 Direct Planning

Under the direct approach, the kinodynamic planning problem becomes much harder because kinematic and dynamic constraints are taken into account simultaneously. Moreover, the planning has to be done in the state space, whose dimension is twice that of the C-space. Existing techniques can be grouped into dynamic programming, optimization, and sampling-based methods.

Dynamic programming approaches search for a solution using a grid of cost-to-go values defined over the state space [1, 23, 67]. The main advantage of this approach is that it can find an optimal solution at the resolution of the grid. Such an approach, however, does not scale well to problems with many degrees of freedom, as the size of the grid grows exponentially in the dimension of the state space.

In contrast, trajectory optimization techniques can be applied to remarkably-complex problems. They aim to find a trajectory $\boldsymbol{x}(t), \boldsymbol{u}(t)$ that minimizes a cost function—generally the cost of executing the trajectory—subject to a set of constraints including the system dynamics. Given an initial condition, $\boldsymbol{x}_s$, and an input trajectory $\boldsymbol{u}(t)$ defined over a finite interval, $t \in [t_0, t_f]$, the basic problem can be formulated as follows:

$$\begin{aligned}
\underset{\boldsymbol{u}(t)}{\text{minimize}} \quad & \int_{t_0}^{t_f} c(\boldsymbol{x}(t), \boldsymbol{u}(t))dt \\
\text{subject to} \quad & \forall t, \dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \\
& \boldsymbol{x}(t_0) = \boldsymbol{x}_s, \\
& \boldsymbol{x}(t_f) = \boldsymbol{x}_g.
\end{aligned}$$

An advantage of this approach is that constraints of any kind can be added to the previous problem. Its drawbacks are that efficient methods to solve this optimization problem may converge to local minima depending on the initial guess employed, and that the problem size becomes huge for long time horizons or systems with many degrees of freedom [73]. Existing trajectory optimization techniques can be classified into transcription and collocation methods [7].

- Transcription methods [50, 81, 96] discretize the trajectory into multiple knot points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$, and then enforce the integral of the dynamics between these points as a constraint:

$$\begin{aligned}
\underset{\boldsymbol{x}_0,\ldots,\boldsymbol{x}_N,\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{N-1}}{\text{minimize}} \quad & T_s \sum_{n=0}^{N-1} c(\boldsymbol{x}_n, \boldsymbol{u}_n) \\
\text{subject to} \quad & \boldsymbol{x}_{n+1} = \boldsymbol{g}_d(\boldsymbol{x}_n, \boldsymbol{u}_n) \quad \forall n \in [0, N-1], \\
& \boldsymbol{x}_0 = \boldsymbol{x}_s, \\
& \boldsymbol{x}_N = \boldsymbol{x}_g.
\end{aligned}$$

  Here, T_s is the time increment employed, and $\boldsymbol{g}_d(\cdot)$ is a discrete approximation of the differential equation, either using an Euler method, or any higher-order method if more accuracy is necessary. Clearly, there is a trade-off between both the number of knot points and the integration method adopted, and the computational cost required to solve the resulting optimization problem.

- Collocation methods [36] alleviate this issue by avoiding numerical integration. Both the input $\boldsymbol{u}(t)$ and state $\boldsymbol{x}(t)$ trajectories are approximated explicitly by means of polynomial functions. Specifically, $\boldsymbol{u}(t)$ is described by a first-order polynomial defined by the $\boldsymbol{u}$ values at the knot points, while $\boldsymbol{x}(t)$ is described by an Hermitian spline defined by the $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ values at such points ($\dot{\boldsymbol{x}}$ being computed using the dynamics in Eq. (9)). Finally, a constraint forces the satisfaction of Eq. (9) at the midpoint of the spline, also known as the collocation point. Thus, the optimization problem can be stated as:

$$\begin{aligned}
\underset{\boldsymbol{x}_0,\ldots,\boldsymbol{x}_N,\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{N-1}}{\text{minimize}} \quad & T_s \sum_{n=0}^{N-1} c(\boldsymbol{x}_n, \boldsymbol{u}_n) \\
\text{subject to} \quad & 0 = \boldsymbol{h}(\boldsymbol{x}_n, \boldsymbol{u}_n, \boldsymbol{x}_{n+1}, \boldsymbol{u}_{n+1}) \quad \forall n \in [0, N-1], \\
& \boldsymbol{x}_0 = \boldsymbol{x}_s, \\
& \boldsymbol{x}_N = \boldsymbol{x}_g,
\end{aligned}$$

  where $\boldsymbol{h}$ refers to the collocation constraint. This method is powerful enough to be applied to challenging problems involving humanoids [16, 17, 41] and kinematic constraints [79]. In particular, in [79] constraints in the form of Eqs. (1) and (2) are enforced at the knot points and then added to the optimization problem. However, a large set of points are still needed to accurately approximate the constraint manifold and the dimension of the optimization problem increases considerably. Moreover, the kinematic constraints are fulfilled at these intermediate points but not necessarily along the whole trajectory. In some applications this might be acceptable, but in our constrained systems it would result in unwanted link penetrations, disassemblies, or contact losses.

A widely-used alternative to dynamic programming and optimization is to rely on kinodynamic sampling-based methods [24, 61]. These methods can cope with high-dimensional problems, and are probabilistically complete. Moreover, recent methods can even generate globally optimal trajectories [37, 52, 53, 65]. The kinodynamic RRT [63] and EST [43] methods stand out among them, due to their effectiveness and conceptual simplicity, since they only require forward motion simulation. However, it is well known that these planners can be inefficient in certain scenarios [20]. Part of the complexity arises from planning in the state space instead of in the lower-dimensional C-space [73]. The RRT method is easier to implement, but its main issue is the disagreement of the metric used to measure the distance between two states, and the actual cost of moving between such states, which must comply with the vector fields defined by Eq. (9) [21, 22, 47, 51, 57, 75, 84, 88]. However, none of the previous methods can directly deal with the implicitly-defined state spaces given by Eq. (5).

6.2 Motion Simulation

Most motion planning approaches heavily rely on the motion simulation of the robotic system. Such simulation presents a big challenge for constrained systems, as their dynamics need to be modeled by differential-algebraic equations [70]. The algebraic equations correspond to Eqs. (1) and (4), which force the robot's state to lie on a manifold embedded in a higher dimensional space; and the differential ones reflect the system dynamics in Eq. (9), which, for each $\boldsymbol{u}$, define a vector field on such manifold. One might think that the integration of Eq. (9) alone using

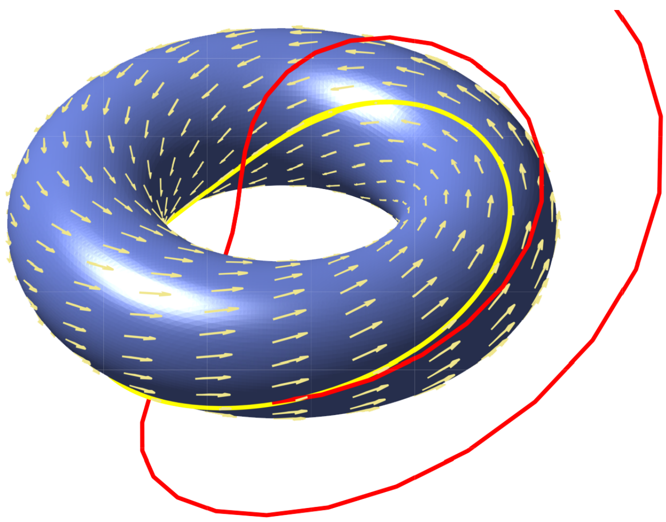


Figure 3: A particle moving on a torus (kinematic constraint) under the shown vector field (dynamic constraint). The trajectory obtained by numerical integration of the vector field (red) increasingly diverges from the exact trajectory (yellow).

standard methods for ordinary differential equations would suffice to predict the motion of the system. However, the solution obtained would easily drift from the algebraic manifold (Fig. 3). Thus, different strategies should also be considered in order to avoid such drift.

Several techniques have been used to this end [3, 60]. In the popular Baumgarte method the drift is alleviated with control techniques [4]. The constraint in Eq. (12) is replaced with a stabilized form, which resembles a damped spring closing the kinematic loop. This method, though, produces artificial energy dissipation and does not work well for relatively complicated systems. Moreover, the control parameters are problem-dependent and no general method to tune them has been given so far [9].

Another way to reduce the drift is to use violation suppression techniques [8, 9, 14]. The constraint error is reduced at each integration step in the directions orthogonal to the constraint manifold without altering the system dynamics. This method, however, also produces energy losses in the system.

A more accurate and totally different approach is given by variational integration methods [92]. In these methods, the integrator is derived by discretizing the variational formulation of a given problem rather than the differential equations. The algorithm preserves the energy of the system and leads to a drift-free integration when simulating constrained systems [49, 64].

The methods relying on local coordinates [33] also produce exact simulations, since they cancel the drift to machine precision. These methods repeatedly express Eq. (9) using local parametrizations of $\mathcal{X}$, integrate the equation in the domain of such parametrizations, and then transform the result back to $\mathcal{X}$. There are several choices for the mentioned parametrizations, including those based on generalized coordinate partitioning [91], exponential maps [68], or tangent space projections [80], to name a few. The latter are particularly interesting because they can be combined with systematic [40] or randomized [45, 46, 76, 77] continuation methods for exploring higher-dimensional manifolds. We next outline how this combination can be exploited to develop a kinodynamic planner for constrained systems.

7 Methodology

Our planner will be based on sampling-based techniques, as these generally work well in high-dimensional problems. Moreover, they can easily cope with the many constraints of the problem, and with any integration method in principle. In particular, we envisage the extension of the classic kinodynamic RRT [63] and EST [43] methods to also deal with constrained robotic systems. To have an idea, we next see how this extension can be achieved with the method in [63]. The extension of [43] would be similar, only requiring adjustments in the expansion heuristics employed.

The planner in [63] assumes that $\mathcal{X}$ is parametrically defined, i.e., that all tuples $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})$ are possible in principle. The planner looks for the desired trajectories $\boldsymbol{u}(t)$ and $\boldsymbol{x}(t)$ by constructing an exploration RRT over $\mathcal{X}$, which in this case is $\mathbb{R}^{2n_q}$. The RRT is rooted at $\boldsymbol{x}_s$ and it is grown incrementally towards $\boldsymbol{x}_g$ while staying inside $\mathcal{X}_{\text{feas}}$. Every tree node stores a feasible state $\boldsymbol{x} \in \mathcal{X}_{\text{feas}}$, and every edge stores the action $\boldsymbol{u} \in \mathcal{U}$ needed to move between the connected states. This action is assumed to be constant during the move. The expansion of the RRT proceeds by applying three steps repeatedly (Fig. 4, top-left). First, a state $\boldsymbol{x}_{rand} \in \mathcal{X}$ is randomly selected; then, the RRT state $\boldsymbol{x}_{near}$ that is closest to $\boldsymbol{x}_{rand}$ is computed according to some metric; finally, a movement from $\boldsymbol{x}_{near}$ towards $\boldsymbol{x}_{rand}$ is performed by applying an action $\boldsymbol{u} \in \mathcal{U}$ during a fixed time Δt. The movement from $\boldsymbol{x}_{near}$ towards $\boldsymbol{x}_{rand}$ is simulated by integrating Eq. (6) numerically, which yields a new state $\boldsymbol{x}$ that may or may not be in $\mathcal{X}_{\text{feas}}$. In the former case $\boldsymbol{x}$ is added to the RRT, and in the latter it is discarded. To test whether $\boldsymbol{x} \in \mathcal{X}_{\text{feas}}$, $\boldsymbol{x}$ is checked

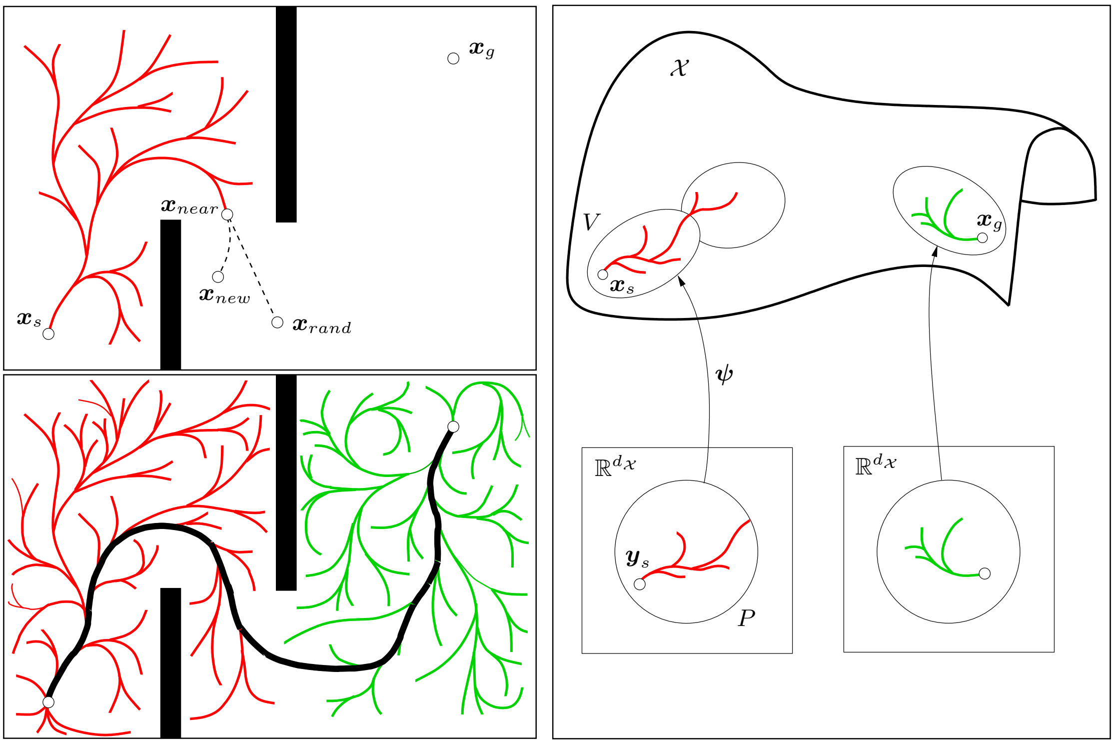


Figure 4: Left-Top: Extension process of an RRT. Left-Bottom: A kinodynamic planning problem is often solved faster with a bidirectional RRT. Right: Construction of an RRT on an implicitly-defined state space manifold.

for collisions by using standard algorithms [48], and the joint positions, velocities and constraint forces are computed to check whether they stay within bounds. The action $\boldsymbol{u}$ applied is typically chosen as the one from $\mathcal{U}$ that brings the robot closer to $\boldsymbol{x}_{rand}$. One can either try all possible values in $\mathcal{U}$ (if it is a discrete set) or only those of n_s random points on $\mathcal{U}$ (if it is continuous). To force the RRT to extend towards $\boldsymbol{x}_g$, $\boldsymbol{x}_{rand}$ is set to $\boldsymbol{x}_g$ once in a while, stopping the whole process when a RRT leaf is close enough to $\boldsymbol{x}_g$. Usually, however, a solution trajectory can be found more rapidly if two RRTs respectively rooted at $\boldsymbol{x}_s$ and $\boldsymbol{x}_g$ are grown simultaneously towards each other (Fig. 4, left-bottom). The expansion of the tree rooted at $\boldsymbol{x}_g$ is based on the integration of Eq. (6) backward in time.

The previous strategy is easy to implement when $\mathcal{X} \in \mathbb{R}^{2n_q}$, but in our case $\mathcal{X}$ is a $d_\mathcal{X}$-dimensional manifold defined by Eqs. (1) and (4). This complicates matters substantially, because there is no straightforward way to randomly select points $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})$ satisfying Eqs. (1) and (4), and the numerical integration of Eq. (6) easily drifts away from $\mathcal{X}$ when standard methods for ordinary differential equations are used. These two issues can be circumvented by constructing an atlas of $\mathcal{X}$ in parallel to the RRT.

An atlas is a collection of charts mapping $\mathcal{X}$ entirely, where each chart is a local diffeomorphism $\boldsymbol{\psi}$ from an open set $P \subseteq \mathbb{R}^{d_\mathcal{X}}$ of parameters to an open set $V \subset \mathcal{X}$ (Fig. 4, right). The V sets can be thought of as partially-overlapping tiles covering $\mathcal{X}$, in such a way that every $\boldsymbol{x} \in \mathcal{X}$

lies in at least one set V. Assuming that an atlas is available, the problem of sampling $\mathcal{X}$ boils down to generating random values $\boldsymbol{y}$ in the P sets, since these values can always be projected to $\mathcal{X}$ using $\boldsymbol{x} = \boldsymbol{\psi}(\boldsymbol{y})$. Also, the atlas allows the conversion of the vector field defined on $\mathcal{X}$ by Eq. (6) into one in the coordinate spaces P, which permits the integration of Eq. (6) using local coordinates [80]. As a result, the RRT motions satisfy Eqs. (1) and (4) by construction, eliminating any drift from $\mathcal{X}$ to machine precision.

One could build a full atlas of the implicitly-defined state space and then use its local parameterizations to define a kinodynamic RRT. However, the construction of a complete atlas is only feasible for low-dimensional state spaces. Moreover, only part of the atlas is necessary to solve a given motion planning problem. Thus, a better alternative is to combine the construction of the atlas and the expansion of the RRT. In this approach, a partial atlas is used to generate random states and to add branches to the RRT. The atlas is initialized with two charts covering $\boldsymbol{x}_s$ and $\boldsymbol{x}_g$, respectively (Fig. 4, right). Then, these charts are used to pull the expansion of the RRT, which in turn adds new charts to the atlas as needed, until $\boldsymbol{x}_s$ and $\boldsymbol{x}_g$ become connected.

8 Resources and Work Plan

The proposed research is framed within the R&D Project "RobCab: Control strategies for cable-driven robot for low-gravity simulation" ($DPI2014-57220-C2-2-P$) of the Spanish Ministry of Science and Innovation, and is partially funded by an FPI grant associated with such project. The work will be developed at the Institut de Robòtica i Informàtica Industrial, UPC-CSIC, in Barcelona, with the Kinematics and Robot Design research group [44].

The algorithms in this research should not require computers with special capabilities. In principle, a standard desktop computer would suffice to solve the motion planning problems confronted. Furthermore, the algorithm will be accompanied by a proof of concept in simulation and in a real robot. The robots available at the Kinematics and Robot Design Lab, or the ones to be constructed within the mentioned research project, will be used to this end.

The work plan for the proposed research is divided into five main tasks, two of which are subdivided into several subtasks, as described below. The schedule of this plan spans over four years and is presented in Fig. 5 as a Gantt chart. In this chart, **Q1**, ..., **Q4** stand for the four quarters of a year, and the work already completed is shown shaded in orange.

Task 1: Literature review

This task entails acquiring a view of the state of the art in kinodynamic motion planning and simulation. Much effort will be devoted to this task at the beginning the thesis in order to put our research in context, but literature review will be a continuous process along this Ph.D. work.

Task 2: Planning strategies

This task encompasses the generalization of the main two kinodynamic planning techniques, the RRT and EST methods, to also deal with constrained robotic systems. The performance of the resulting methods will be compared in the end.

Task 2.1: Planning based on RRTs

The RRT method will be generalized by incrementally building an atlas of the implicitly-defined state space $\mathcal{X}$. This atlas can then be used to generate random samples needed to expand the exploration tree. Moreover, the dynamics of the constrained system, modeled with differential-algebraic equations, can be accurately integrated using the atlas [80].

One of the main issues to consider is the metric used to measure the distance between two states. This is a general issue of all sampling-based kinodynamic planners [84], but in our context it is harder since the metric should also consider the vector fields defined by the dynamic constraints, and the curvature of the state space manifold defined by the kinematic equations.

Task 2.2: Planning based on ESTs

The EST method will also be generalized by building an atlas of $\mathcal{X}$. Here, however, we aim to estimate the local density of neighboring nodes by exploiting such atlas. Then, this density will be used to expand the tree towards unexplored areas of $\mathcal{X}$. In this task it will also be necessary to develop heuristics to bias the expansion of the tree towards the goal state [74].

Task 3: Extensions

Several extensions to the sampling-based approaches should be included in order to make the planner as general as possible.

Task 3.1: Constraint force bounds

This task encompasses the computation of constraint forces in order to ensure they stay within bounds. This is necessary in applications involving cable-driven robots, for example, in which the cable tensions should remain positive for a proper operation.

Task 3.2: Virtual constraints

Constraints that are imposed by the task and not by the robot structure will also be treated by the planner. One can compute the constraint forces that restrict the robot motion to fulfill virtual constraints. Such forces should then be applied by the actuators.

Task 3.3: Non-holonomic constraints

This task aims to include the non-holonomic constraints in Eq. (3) to the planning problem. Due to such constraints, the navigation of the state-space manifold is more restricted, which complicates the planning. Nonetheless, these constraints can be included in Eq. (13) and then, the motion of the constrained robot can be simulated by using Eq. (14), as long as state-space singularities are not present or actively avoided.

Task 3.4: Trajectory smoothing

Due to the randomness of all sampling-based methods, the input actions are not continuous and may lead to nonsmooth solution trajectories. Therefore, this task encompasses the smoothing of the resulting position, velocity and acceleration trajectories. Such smoothing should be done in $\mathcal{X}$ and thus, we might exploit the constructed atlas to this end.

Task 3.5: Cost reduction

This task aims to reduce the initial cost of the planned trajectory. In this sense, locally optimal trajectories could be obtained by feeding the output of the planner into optimization approaches [18]. Globally optimal ones could also be generated by considering the trajectory cost right into the planner [37, 65].

Task 4: Implementation and validation

This task consists in the implementation and validation of the planning algorithms developed in previous tasks. On this regard, a main task will be the selection of a proper formulation for the dynamic equations of the robot. The use of the joint variables as generalized coordinates seems to be a good choice [30], since its facilitates the modeling of actuator forces and friction effects. Moreover, the use of recursive algorithms to efficiently obtain the various magnitudes involved in the system dynamics is proving to be effective [30]. Therefore, in this task we expect to implement these algorithms for the motion simulation in the CUIK Suite [56].

Finally, throughout the different tasks, the planner will be tested in simulation environments. In the last stages of the thesis, however, it is expected to apply the kinodynamic planner to a real robot. To this end, a standard controller will be implemented to follow the nominal trajectory obtained from the planner.

Task 5: Dissertation writing and defence

This last task entails the elaboration of the dissertation and the preparation of its public defence.

9 Publications

The following papers have been prepared to disseminate the research results obtained so far. The first paper describes the RRT planner in Task 2.1. The second paper is related to Task 3.1, as it is an application of the RRT planner to cable-driven robots, where it is shown how constraint forces can be computed to ensure positive tension in the cables.

Conferences

C1. Bordalba, R., Ros, L., and Porta, J.M. Kinodynamic Planning on Constraint Manifolds. In *Robotics: Science and Systems 2017. Submitted.*

C2. Bordalba, R., Ros, L., and Porta, J.M. Collision-free Kinodynamic Planning for Cable-suspended Parallel Robots. In *CableCon 2017: Third International Conference on Cable-Driven Parallel Robots. Submitted.*

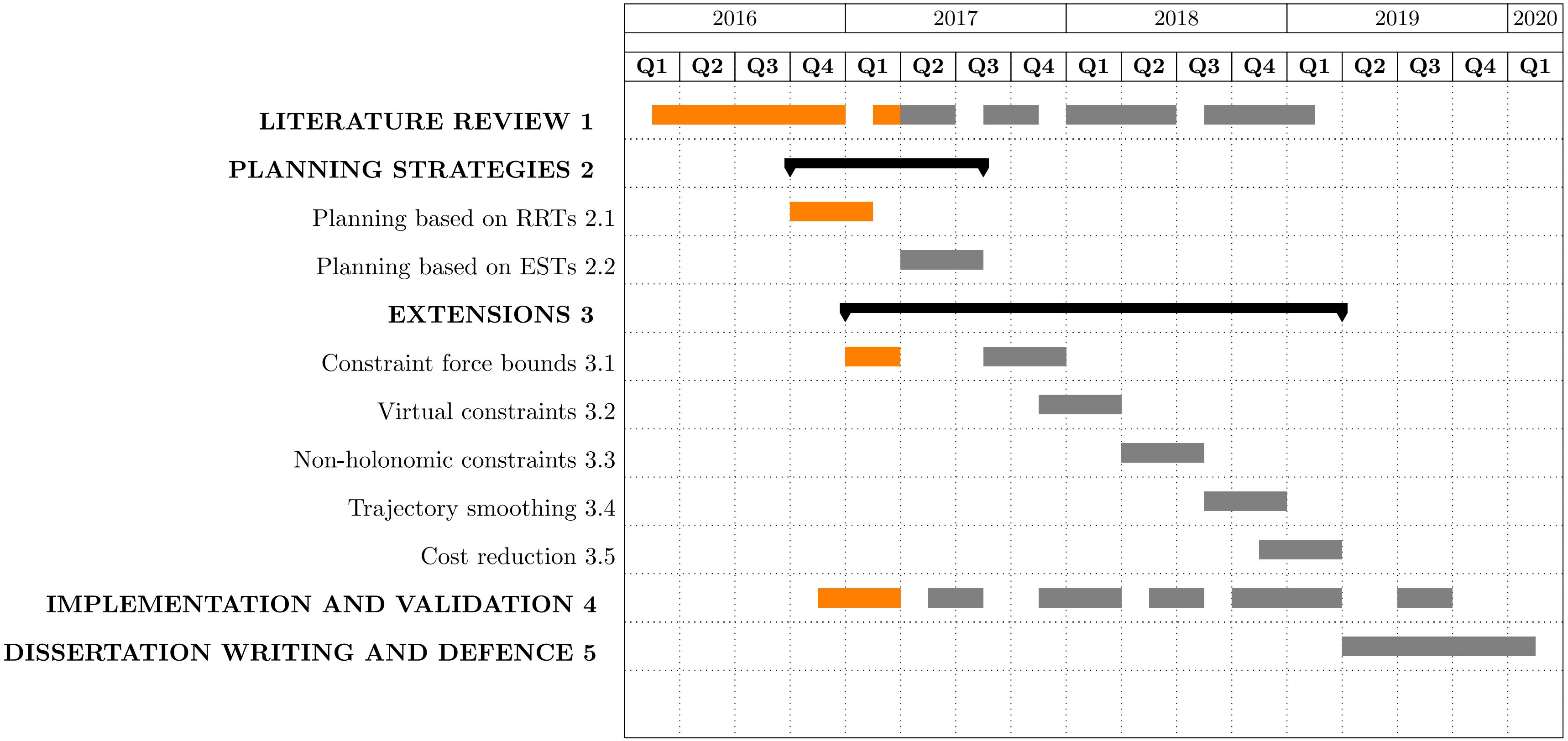


Figure 5: Work plan of the proposed work

References

[1] J. Barraquand and J.-C. Latombe. Nonholonomic multibody mobile robots: controllability and motion planning in the presence of obstacles. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 2328–2335, 1991.

[2] J. Barraquand and J. C. Latombe. Robot Motion Planning: A Distributed Representation Approach. *International Journal of Robotics Research*, 10(6):628–649, 1991.

[3] O. A. Bauchau and A. Laulusa. Review of contemporary approaches for constraint enforcement in multibody systems. *Journal of Computational and Nonlinear Dynamics*, 3(1): 011005, 2008.

[4] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1(1):1–16, 1972.

[5] D. Berenson, S. Srinivasa, D. Ferguson, and J. Kuffner. Manipulation planning on constraint manifolds. In *IEEE International Conference on Robotics and Automation*, pages 1383–1390, 2009.

[6] D. Berenson, S. Srinivasa, and J. J. Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *International Journal of Robotics Research*, 30(12): 1435–1460, 2011.

[7] J. T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[8] W. Blajer. Elimination of constraint violation and accuracy aspects in numerical simulation of multibody systems. *Multibody System Dynamics*, 7(3):265–284, 2002.

[9] W. Blajer. Methods for constraint violation suppression in the numerical simulation of constrained multibody systems – A comparative study. *Computer Methods in Applied Mechanics and Engineering*, 200(13-16):1568–1576, 2011.

[10] J. E. Bobrow, S. Dubowsky, and J. S. Gibson. Time-optimal control of robotic manipulators along specified paths. *International Journal of Robotics Research*, 4(3):3–17, 1985.

[11] O. Bohigas, M. E. Henderson, L. Ros, M. Manubens, and J. M. Porta. Planning singularity-free paths on closed-chain manipulators. *IEEE Transactions on Robotics*, 29(4):888–898, 2013.

[12] O. Bohigas, M. Manubens, and L. Ros. *Singularities of robot mechanisms: numerical computation and avoidance path planning*, volume 41 of *Mechanisms and Machine Science*. Springer, 2016.

[13] R. Bohlin and L. E. Kavraki. Path planning using lazy PRM. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 521–528, 2000.

[14] D. J. Braun and M. Goldfarb. Eliminating constraint drift in the numerical simulation of constrained dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 198(37–40):3151–3160, 2009.

[15] R. A. Brooks and T. Lozano-Pérez. A subdivision algorithm in configuration space for findpath with rotation. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(2): 224–233, March 1985. ISSN 0018-9472. doi: 10.1109/TSMC.1985.6313352.

[16] T. Buschmann, S. Lohmeier, M. Bachmayer, H. Ulbrich, and F. Pfeiffer. A collocation method for real-time walking pattern generation. In *International Conference on Humanoid Robots*, pages 1–6. IEEE, 2007.

[17] T. Buschmann, S. Lohmeier, and H. Ulbrich. Humanoid robot lola: Design and walking control. *Journal of physiology-Paris*, 103(3):141–148, 2009.

[18] S. D. Butler, M. Moll, and L. E. Kavraki. A general algorithm for time-optimal trajectory generation subject to minimum and maximum constraints. In *Workshop on the Algorithmic Foundations of Robotics*, 2016.

[19] J. Canny. *The Complexity of Robot Motion Planing.* MIT Press, 1988.

[20] P. Cheng. *Sampling-based motion planning with differential constraints.* Phd dissertation, Department of Computer Science, University of Illinois, 2005.

[21] P. Cheng and S. M. LaValle. Reducing metric sensitivity in randomized trajectory design. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 43–48, 2001.

[22] P. Cheng and S. M. LaValle. Resolution complete rapidly-exploring random trees. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 267–272, 2002.

[23] M. Cherif. Kinodynamic motion planning for all-terrain wheeled vehicles. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 317–322, 1999.

[24] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations.* Intelligent Robotics and Autonomous Agents. MIT Press, 2005.

[25] J. Cortés and T. Siméon. Sampling-Based Motion Planning under Kinematic Loop Closure Constraints. *6th International Workshop on the Algorithmic Foundations of Robotics*, pages 75–90, 2004.

[26] S. Dalibard, A. Nakhaei, F. Lamiraux, and J.-P. Laumond. Whole-body task planning for a humanoid robot: a way to integrate collision avoidance. In *IEEE-RAS International Conference on Humanoid Robots*, pages 355–360, 2009.

[27] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959. ISSN 0945-3245. doi: 10.1007/BF01386390. URL `http://dx.doi.org/10.1007/BF01386390`.

[28] B. Donald, P. Xavier, J. Canny, and J. Reif. Kinodynamic motion planning. *Journal of the ACM*, 40(5):1048—-1066, 1993.

[29] S. Dubowsky and Z. Shiller. Optimal dynamic trajectories for robotic manipulators. In *IFToMM Symposium on Robot Design, Dynamics and Control*, pages 133–143, 1985.

[30] Roy Featherstone. *Rigid body dynamics algorithms.* Springer, 2014.

[31] M. Gharbi, J. Cortés, and T. Siméon. A sampling-based path planner for dual-arm manipulation. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 383–388, 2008.

[32] D. T. Greenwood. *Advanced dynamics.* Cambridge University Press, 2006.

[33] E. Hairer. Geometric integration of ordinary differential equations on manifolds. *BIT Numerical Mathematics*, 41(5):996–1007, 2001.

[34] L. Han and N. M. Amato. A kinematics-based probabilistic roadmap method for closed chain systems. In *Algorithmic and Computational Robotics - New Directions*, pages 233–246, 2000.

[35] L. Han, L. Rudolph, J. Blumenthal, and I. Valodzin. Convexly stratified deformation spaces and efficient path planning for planar closed chains with revolute joints. *International Journal of Robotics Research*, 27(11-12):1189–1212, 2008.

[36] C. R. Hargraves and S. W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987.

[37] K. Hauser and Y. Zhou. Asymptotically optimal planning by feasible kinodynamic planning in a state-cost space. *IEEE Transactions on Robotics*, 32(6):1431–1443, 2016.

[38] K. Hauser, T. Bretl, J. C. Latombe, K. Harada, and B. Wilcox. Motion planning for legged robots on varied terrain. *International Journal of Robotics Research*, 27(11-12):1325–1349, 2008.

[39] I. Havoutis and S. Ramamoorthy. Motion synthesis through randomized exploration of submanifolds of configuration spaces. In *RoboCup 2009: Robot Soccer World Cup XIII. Lecture Notes in Artificial Intelligence*, volume 5949, pages 92–103, 2009.

[40] M. E. Henderson. Multiple parameter continuation: computing implicitly defined k-manifolds. *International Journal of Bifurcation and Chaos*, 12(3):451–476, 2002.

[41] A. Hereid, E. A. Cousineau, C. M. Hubicki, and A. D. Ames. 3D dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics. In *IEEE International Conference on Robotics and Automation*, pages 1447–1454. IEEE, 2016.

[42] D. Hsu, J. C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 2719–2726. IEEE, 1997.

[43] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock. Randomized kinodynamic motion planning with moving obstacles. *International Journal of Robotics Research*, 21(3):233–255, 2002.

[44] Institut de Robòtica i Informàtica Industrial, UPC-CSIC. `http://www.iri.upc.edu/`.

[45] L. Jaillet and J. M. Porta. Path planning under kinematic constraints by rapidly exploring manifolds. *IEEE Transactions on Robotics*, 29(1):105–117, 2013.

[46] L. Jaillet and J.M. Porta. Path planning with loop closure constraints using an atlas-based RRT. *International Symposium on Robotics Research*, 2011.

[47] L. Jaillet, J. Hoffman, J. van den Berg, P. Abbeel, J. M. Porta, and K. Goldberg. EG-RRT: Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2646–2652, 2011.

[48] P. Jiménez, F. Thomas, and C. Torras. 3D collision detection: A survey. *Computers & Graphics*, 25(2):269–285, 2001.

[49] E. R. Johnson and T. D. Murphey. Scalable variational integrators for constrained mechanical systems in generalized coordinates. *IEEE Transactions on Robotics*, 25(6):1249–1261, 2009.

[50] 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*, pages 4569–4574, 2011.

[51] M. Kalisiak. *Toward more efficient motion planning with differential constraints*. PhD thesis, Computer Science,University of Toronto, 2008.

[52] S. Karaman and E. Frazzoli. Optimal kinodynamic motion planning using incremental sampling-based methods. In *IEEE Conference on Decision and Control*, pages 7681–7687. IEEE, 2010.

[53] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller. Anytime motion planning using the RRT*. In *IEEE International Conference on Robotics and Automation*, pages 1478–1483, 2011.

[54] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[55] O. Khatib. Real-time Obstacle Avoidance for Manipulators and Mobile Robots. *International Journal of Robotics Research*, 5(1):90–98, April 1986. ISSN 0278-3649.

[56] KRD Group. The CuikSuite software. `http://www.iri.upc.edu/cuik`, 2012.

[57] A. M. Ladd and L. E. Kavraki. Fast tree-based exploration of state space for robots with dynamics. In *Algorithmic Foundations of Robotics VI*, pages 297–312, 2005.

[58] F. Lamiraux, E. Ferré, and E. Vallée. Kinodynamic motion planning: Connecting exploration trees using trajectory optimization methods. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3987–3992. IEEE, 2004.

[59] J.-C. Latombe. *Robot Motion Planning*. Engineering and Computer Science. Springer, 1991.

[60] A. Laulusa and O. A. Bauchau. Review of classical approaches for constraint enforcement in multibody systems. *Journal of Computational and Nonlinear Dynamics*, 3(1):011004, 2008.

[61] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, New York, 2006.

[62] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and Computational Robotics - New Directions*, pages 293–308, 2000.

[63] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, 2001.

[64] S. Leyendecker, J. E. Marsden, and M. Ortiz. Variational integrators for constrained dynamical systems. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 88(9):677–708, 2008.

[65] Y. Li, Z. Littlefield, and K. E. Bekris. Asymptotically optimal sampling-based kinodynamic planning. *International Journal of Robotics Research*, 35(5):528–564, 2016.

[66] H-H. Lu. Computation of Path Constrained Time Optimal lotions With Dynamic Singularities. *Journal of Dynamic Systems, Measurement, and Control*, 114:34–40, 1992. doi: 10.1115/1.2896505.

[67] K. M. Lynch and M. T. Mason. Stable pushing: mechanics, controllability, and planning. *International Journal of Robotics Research*, 15(6):533–556, 1996.

[68] H. Munthe-Kaas. High order Runge-Kutta methods on manifolds. *Applied Numerical Mathematics*, 29(1):115–127, 1999.

[69] J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. In *Algorithmic Foundations of Robotics V*, pages 221–237. Springer, 2004.

[70] L. R. Petzold. Numerical solution of differential-algebraic equations in mechanical systems simulation. *Physica D: Nonlinear Phenomena*, 60(1–4):269–279, 1992.

[71] F. Pfeiffer and R. Johanni. A concept for manipulator trajectory planning. *IEEE Journal on Robotics and Automation*, 3(2):115–123, 1987.

[72] Q. C. Pham and O. Stasse. Time-optimal path parameterization for redundantly actuated robots: A numerical integration approach. *IEEE/ASME Transactions on Mechatronics*, 20 (6):3257–3263, 2015.

[73] Q.-C. Pham, S. Caron, P. Lertkultanon, and Y. Nakamura. Admissible velocity propagation: Beyond quasi-static path planning for high-dimensional robots. *International Journal of Robotics Research*, 36(1):44–67, 2017.

[74] J. M. Phillips, N. Bedrossian, and L. E. Kavraki. Guided expansive spaces trees: A search strategy for motion-and cost-constrained state spaces. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3968–3973. IEEE, 2004.

[75] E. Plaku, L. Kavraki, and M. Vardi. Discrete search leading continuous exploration for kinodynamic motion planning. In *Proceedings of Robotics: Science and Systems*, 2007.

[76] J. M. Porta and L. Jaillet. Path planning on manifolds using randomized higher-dimensional continuation. *Workshop on the Algorithmic Foundations of Robotics*, pages 337–353, 2010.

[77] J. M. Porta, L. Jaillet, and O. Bohigas. Randomized path planning on manifolds based on higher-dimensional continuation. *International Journal of Robotics Research*, 31(2):201–215, 2012.

[78] J. M. Porta, L. Ros, O. Bohigas, M. Manubens, C. Rosales, and L. Jaillet. The Cuik Suite: Analyzing the motion closed-chain multibody systems. *IEEE Robotics and Automation Magazine*, 21(3):105–114, 2014.

[79] M. Posa, S. Kuindersma, and R. Tedrake. Optimization and stabilization of trajectories for constrained dynamical systems. In *IEEE International Conference on Robotics and Automation*, pages 1366–1373, 2016.

[80] F. A. Potra and J. Yen. Implicit numerical integration for Euler-Lagrange equations via tangent space parametrization. *Journal of Structural Mechanics*, 19(1):77–98, 1991.

[81] J. Schulman, Duan Y, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel. Motion planning with sequential convex optimization and convex collision checking. *International Journal of Robotics Research*, 33(9):1251–1270, 2014.

[82] K. Shin and N. McKay. Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions on Automatic Control*, 30(6):531–541, 1985.

[83] A. Shkolmik and R. Tedrake. Path planning in 1000+ dimensions using a task-space Voronoi bias. In *IEEE International Conference on Robotics and Automation*, pages 2892–2898, 2009.

[84] A. Shkolnik, M. Walter, and R. Tedrake. Reachability-guided sampling for planning under differential constraints. In *IEEE International Conference on Robotics and Automation*, pages 2859–2865, 2009.

[85] N. Shvlab, G. Liu, M. Shoham, and J. C. Trinkle. Motion planning for a class of planar closed-chain manipulators. *International Journal of Robotics Research*, 26(5):457–473, 2007.

[86] J. J. E. Slotine and H. S. Yang. Improving the efficiency of time-optimal path-following algorithms. *IEEE Transactions on Robotics and Automation*, 5(1):118–124, 1989.

[87] M. Stilman. Task constrained motion planning in robot joint space. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3074–3081, 2007.

[88] I. A. Şucan and L. E. Kavraki. A sampling-based tree planner for systems with complex dynamics. *IEEE Transactions on Robotics*, 28(1):116–131, 2012.

[89] C. Suh, T. T. Um, B. Kim, H. Noh, M. Kim, and F. C. Park. Tangent space RRT: A randomized planning algorithm on constraint manifolds. In *IEEE International Conference on Robotics and Automation*, pages 4968–4973, May 2011. doi: 10.1109/ICRA.2011.5980566.

[90] X. Tang, S. Thomas, P. Coleman, and N. M Amato. Reachable distance space: efficient sampling-based planning for spatially constrained systems. *International Journal of Robotics Research*, 29(7):916–934, 2010.

[91] R. A. Wehage and E. J. Haug. Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. *Journal of mechanical design*, 104(1):247–255, 1982.

[92] M. West. *Variational integrators*. PhD thesis, California Institute of Technology, 2004.

[93] J. H. Yakey, S. M. LaValle, and L. E. Kavraki. Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation*, 17(6):951–959, 2001.

[94] Z. Yao and K. Gupta. Path planning with general end-effector constraints: using task space to guide configuration space search. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1875–1880, 2005.

[95] A. Yershova and S. M. LaValle. Motion planning for highly constrained spaces. In *Robot Motion and Control. Lecture Notes on Control and Information Sciences*, volume 396, pages 297–306, 2009.

[96] 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. *International Journal of Robotics Research*, 32(9-10):1164–1193, 2013.