Safety-Aware Optimal Control in Motion Planning

Han Wang[1], Xuda Ding[2], Jianping He[2], Kostas Margellos[1], Antonis Papachristodoulou[1]

***Abstract*—Optimisation is a popular tool of generating control laws for motion planning in a complex environment. The existence of multiple irregular obstacles in the working space introduces nonconvex collision avoidance constraints, making the optimal control problem intractable. One efficient approach to address this issue is successive convex approximation (SCA), where the nonconvex problem is convexified and solved successively. However, this approach still faces two main challenges: i) high computational complexity, incurred by multiple constraints when solving the optimal control problem with a long planning horizon and multiple obstacles; ii) infeasibility, caused by linearization about infeasible reference points. To address these challenges, this paper proposes a backward receding SCA (BRSCA) approach, which leverages dynamic programming with a primal-dual iteration to decrease computational complexity; and designs a dynamic constraints-selection rule to avoid infeasibility. Moreover, it is found that the proposed BRSCA approach is applicable to time-varying control limits. Numerical simulations and hardware experiments demonstrate that BRSCA has a higher probability of finding feasible solutions and reduces the computation time by at least 17.4%, compared to other methodologies in the literature.**

I. Introduction

The motion planning problem is an important research topic in robotics. Existing approaches for the problem can be grouped into two categories, i.e., graph search approaches and trajectory optimisation approaches. Graph search approaches [1] [2] are widely used in 2-D scenarios, through discretizing the working space and efficiently finding collision-free trajectories. When applied to high dimensional spaces, these approaches often encounter real-time problems. Trajectory optimisation methodologies scale well with the state space dimension. However, the nonconvex collision avoidance constraints make the formulated optimisation problem nonconvex [3], and hence intractable. Even a feasible solution is hard to be determined with a nonlinear optimisation solver like Sequential Quadratic Programming (SQP). To overcome nonconvexity, two different approaches have been proposed.

The first involves partitioning the safety region into a series of convex regions. It has been proposed to use polyhedra for the region generation, and then the trajectory optimisation was formulated as a mixed-integer programming problem [4] [5]. Polynomials and splines were also considered in more recent works [6] [7]. These approaches are highly efficient for finding a feasible trajectory, while without considering system dynamics and controller design, i.e., stability and optimal energy consumption.

[1]The authors are with the Department of Engineering Science, University of Oxford, Oxford, United Kingdom. E-mails: {han.wang, kostas.margellos, antonis}@eng.ox.ac.uk

[2]The authors are with the Department of Automation Shanghai Jiao Tong University, Shanghai, China. E-mails: {dingxuda, jphe}@sjtu.edu.cn

The second stream of methods gradually convexifies the nonconvex collision avoidance constraints via linearization. This type of approach was firstly proposed for the difference of convex programming (DCP) problems [8]. These approaches split the nonconvex function into the difference of two convex parts, then successfully convexify the constraints' nonconvex parts via linearization about a reference point, thus termed successive convex approximation (SCA) or sequential convex programming (SCA). Variations of such approaches are proposed in the optimisation community in the realm of the convex-concave procedure [9]. This efficient reformulation has been widely used in many recent optimisation-based works, e.g., [10]–[12]. Although mathematically rigorous, the SCA still faces critical issues in real applications. The first one is that a feasible initial guess is required. Slack variables are used to relax this issue [9], but it still lacks a theoretical guarantee. The second issue is that the search space is prone to be empty due to the presence of a high number of constraints. An incremental SCA (iSCA) approach has been proposed, which incrementally includes the violated constraints into the optimisation problem and guarantees a lower computation complexity [13]. However, in some multi-constraint scenarios, the trajectory calculated by iSCA is infeasible. The reason is that the convexification about infeasible reference points possibly renders the convex search spaces infeasible.

To design stable and optimal control laws for safe motion planning, model predictive control (MPC) and optimal control with collision avoidance constraints are widely investigated. Modern nonlinear solvers enable us to reformulate the problem into a generalized optimisation problem with an equality constraint (system dynamics) and inequality constraints (collision avoidance and input limits). The computational complexity of such a problem grows quadratically with the length of the planning horizon, and the number of obstacles [14]. So even a linear quadratic regulator (LQR) model with multiple convexified constraints is challenging. Many efforts have been devoted to solve this problem with multi-parametric optimisation, which partitions the state space and substitutes multiple constrained LQR sub-problems for the original problem [15]–[17]. Recently, [18] proposed a density function-based approach that generates a control law for whole state space, whereas the approach requires solving complicated, ordinary differential equations in every iteration.

Inspired by the previous studies, in this paper we aim to design stable and optimal control laws for safe motion planning with lower computation complexity. This paper proposes a novel primal-dual framework for solving the constrained LQR problem through backward recursion. After convexifying all the nonconvex constraints with the proposed BRSCA scheme, the problem is transformed into a convex quadratically constrained quadratic programming (QCQP). The proposed approach provides an explicit solution for the optimal cost-to-go and the control law. Time-varying control limits are also considered in controller synthesis. Our work provides a solution for the sub-problem in [19]. This work is mostly related to the recent constrained differential dynamic programming approach [20] [21]. Unlike lifting the constraints into the cost with barrier functions, we model the constraints in a hard manner. In addition, our method is especially suitable for linear systems, as the cost-to-go can be accurately modelled. The contributions can be summarized as follows:

- We propose a novel BRSCA approach to deal with infeasible reference point, as well as provide acceleration mechanisms;
- We present a primal-dual framework for solving convex constrained LQR efficiently, with the closed-form solutions for the optimal cost-to-go and the control law;
- We demonstrate higher computational efficiency and success rate against existing methods by means of a detailed numerical study.

The rest of this paper is organized as follows. Section II formulates the optimal control problem with constraints, and provides the definition of a semi-convex function. Backward receding successive convex approximation for convexifying the original problem is presented in Section III. A primal-dual approach is shown in Section IV. Section V presents the numerical simulations and hardware implementations. Section VI concludes the paper and provides some directions for future work.

II. Problem Formulation

We consider the robot motion planning problem with n-dimensional state space. The objective is to synthesize an optimal control law along with an optimal trajectory for a single robot. We assume the robot has a linear and deterministic kinematic model, denoted by

$$x_{t+1} = Ax_t + Bu_t, \tag{1}$$

where $x_t \in \mathcal{X} \subseteq \mathbb{R}^n$ denotes the state, and $u_t \in \mathcal{U}_t \subset \mathbb{R}^m$ denotes the control input. $A \in \mathbb{R}^{n\times n}$ is the state transition matrix, and $B \in \mathbb{R}^{n\times m}$ is the input matrix. Assume that A and B are known. To avoid collision, all states $\{x_0, \dots, x_t, \dots\}$ should be within a safety set $\mathcal{X}$ defined by the zeroing barrier function proposed in [22]. Suppose there exit a series of functions $h_i : \mathbb{R}^n \to \mathbb{R}$:

$$\begin{aligned} \mathcal{X} &= \cap_{i\in\mathcal{I}}\{x \in \mathbb{R}^n | h_i(x) \geq 0\}, \\ \partial\mathcal{X} &= \cup_{i\in\mathcal{I}}\{x \in \mathbb{R}^n | h_i(x) = 0\}, \\ \text{Int}(\mathcal{X}) &= \cap_{i\in\mathcal{I}}\{x \in \mathbb{R}^n | h_i(x) > 0\}, \\ \bar{\mathcal{X}} &= \cup_{i\in\mathcal{I}}\{x \in \mathbb{R}^n | h_i(x) < 0\}, \end{aligned} \tag{2}$$

where $\partial\mathcal{X}$ represents the boundary of safety set, Int($\mathcal{X}$) and $\bar{\mathcal{X}}$ are the interior and complementary set of $\mathcal{X}$, respectively. A sequence of functions $h_1(x), \dots, h_{\mathcal{I}}(x)$ can be used to establish the safety set and obstacle descriptions where every obstacle is specified by an unique index in $\mathcal{I}$. With a slight abuse of notation, let the subscript $\mathcal{I}$ represent the last index. Assume all these sets are smooth, compact, and *semi-convex*.

Definition 1 (*Semi-convex*)**.** *A function h_i is said to be semi-convex if there exits a positive semidefinite matrix $H_i \in \mathbb{R}^{n\times n}$ such that the function $\tilde{h}_i(x)$, satisfying*

$$\tilde{h}_i(x) \triangleq h_i(x) + \boldsymbol{H}_i(x, x_0), \tag{3}$$

is convex for any $x_0 \in \mathbb{R}^n$, where

$$\boldsymbol{H}_i(x, x_0) := \frac{1}{2}(x - x_0)^\top H_i(x - x_0)$$

is a quadratic function with respect to x_0.

We call the sets defined by the semi-convex functions are semi-convex. Semi-convex obstacles are quite common in practice. Roughly speaking, for every feasible state if there exits a convex quadratic closure that covers it without intersecting with the obstacle, we say the function corresponding to the obstacle is *semi-convex*.

Unlike quadratic programming (QP)-based control barrier function approaches, which only depend on the current state, we focus here on transforming the problem into an optimal control problem with collision avoidance constraints at every time instant.

$$\min_{u\in\mathbb{R}^n} \quad J(u) = {x_T}^\top P x_T + \sum_{t=0}^{T-1} {x_t}^\top Q x_t + {u_t}^\top R u_t, \tag{4a}$$

$$s.t. \quad x_{t+1} = Ax_t + Bu_t, t = 0, \dots, T-1, \tag{4b}$$

$$h_i(x_t) \geq 0, t = 1, \dots, T-1, i \in \mathcal{I}, \tag{4c}$$

$$\mathcal{G}_t(u_t) = G_t u_t + e_t \leq 0, t = 0, \dots, T-1, \tag{4d}$$

where $P \succeq 0$ denotes the terminal cost function, $\mathcal{G}_t(u_t) \leq 0$ formulates the time-varying control admissible set $\mathcal{U}_t$ (where $G_t \in \mathbb{R}^{s\times n}, e_t \in \mathbb{R}^s$). Note that the terminal term ${x_T}^\top P x_T$ is used to reach and stabilize around equilibrium. The length of the horizon T depicts the trade-off between computational complexity and conservatism. Some necessary assumptions are given below.

Assumption 1. *All functions $\{-h_1(x), \dots, -h_{\mathcal{I}}(x)\}$ in* (2) *are semi-convex. The robot is aware of the shape expression h_i for every obstacle, $i = 1, \dots, \mathcal{I}$.*

We note here we assume $-h_i(x)$ is *semi-convex* but not $h_i(x)$, since the collision-free constraints can be reformulated as $-h_i(x) \leq 0$.

Assumption 2. *(A, B) is stabilizable, $(A, \sqrt{Q})$ is detectable, $Q \succeq 0$, $R \succ 0$.*

As mentioned in our motivation, our goal is to solve the constrained problem (4) explicitly. The main challenges here are twofold: the constraint set $h_i(x_t) \geq 0$ makes it hard to leverage dynamic programming to construct a backward iterative law, and the constraint function $h_i(x_t)$ is not necessarily convex.

III. Backward Receding Successive Convex Approximation

We presents a variation of Successive Convex Approximation (SCA), termed *backward receding SCA* (BRSCA). This

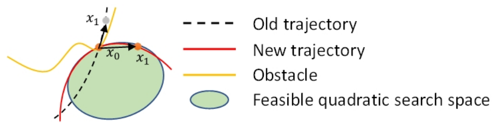

Fig. 1: A demonstration of forming the convex search space and the new trajectory

method is an extension of the incremental SCA (iSCA) [13]. The principle is the same, where we check the feasibility of all constraints, and include only the violated ones into the problem. A backward receding scheme is proposed to cover the infeasible reference points.

A. Successive Convex Approximation

SCA was proposed for nonconvex collision avoidance constraint $h_i(x_0) \geq 0$ about state x_0. The principle of SCA is to split the nonconvex function into the sum of a convex and a concave function, and successively linearize the concave one about the current state. We can obtain the summation of the convex part and the concave part through exploiting semi-convexity. Consider one nonconvex function $h_i(x)$ with reference point x_0. Then the explicit expression of one convexified candidate $\hat{h}_i(x)$ takes the form

$$\hat{h}_i(x) = \underbrace{-h_i(x_0) - \nabla h_i(x_0)^\top (x - x_0)}_{\text{linearized concave part}} + \underbrace{\mathbf{H}_i(x, x_0)}_{\text{convex part}}. \tag{5}$$

Lemma 1. *Consider $\hat{h}_i(x)$ is convex. Then the safe set defined by $\{x | -\hat{h}_i(x) > 0\}$ is a subset of that defined by $\{x | h_i(x) > 0\}$, i.e., $-\hat{h}_i(x) \leq h_i(x), \forall x \in \mathbb{R}^n$.*

Proof. By Definition 1, $h_i(x) + \mathbf{H}_i(x, x_0)$ is convex since $h_i(x)$ is assumed to be *semi-convex*. Hence we can substitute

$$-h_i(x) - \mathbf{H}_i(x, x_0) + \mathbf{H}_i(x, x_0) < 0$$

for the safe constraint $-h_i(x) < 0$. Note that $-h_i(x) - \mathbf{H}_i(x, x_0)$ and $\mathbf{H}_i(x, x_0)$ are concave and convex, respectively. By the definition of convexity, we obtain:

$$h_i(x) - h_i(x_0) \geq -\mathbf{H}_i(x, x_0) + \nabla h_i(x_0)(x - x_0),$$

which implies that

$$h_i(x) \geq h_i(x_0) - \mathbf{H}_i(x, x_0) + \nabla h_i(x_0)(x - x_0) = -\hat{h}_i(x),$$

i.e., $-\hat{h}_i(x) \leq h_i(x)$ holds. Besides, since $\mathbf{H}_i(x, x_0)$ is convex and $-h_i(x) - \nabla h_i(x_0)^\top (x - x_0)$ is affine, $\hat{h}_i(x)$ is convex. □

We obtain the convexified expressions for each constraint through convexification (5). However, when the amount of obstacles is large, the search space can be highly limited, and even feasibility of the problem is not guaranteed.

B. Backward Receding SCA

We propose a further variation to SCA based on a backward receding scheme, shown in Algorithm 1, to enlarge the feasible search space, as well as increase the success rate.

Step 3 of Algorithm 1 is similar to iSCA in that the constraints are included dynamically. Only violated collision avoidance constraints are included in the constrained LQR problem, while we only consider the included constraints in the problem. Steps 4 - 8 of Algorithm 1 show the backward receding based convexification approach. Here we introduce a novel approach of using the convexified search space about the closest backward feasible state, instead of that about an infeasible state. This efficient rule eliminates the feasible initial guess requirement of the iSCA. In step 10 we solve the constrained LQR problem for optimal control law $u^* = \{u_0^*, \ldots, u_{T-1}^*\}$ and obtain the corresponding trajectory, and the optimal cost value $J^*(u^*)$.

Algorithm 1: Backward receding successive convex approximation algorithm

```
Input: length of planning horizon $T$, safe set $\mathcal{X}$, initial start
       point $x_0$, end point $x_T$, trajectory $\{x_0^0, \ldots, x_T^0\}$
       solved from (4) without collision free constraints
Output: optimal control law $u^*$, optimal cost value $J^*(u^*)$.
1  while $\exists i, t,\ h_i(x_t^k) \leq 0$ do
2      include the violated constraints $h_i(x_t^k) \leq 0$.
3      while $J(u^{k+1}) < J(u^k)$ do
4          for all the non-violated constraints $h_j(x_t^k) > 0$ do
5              convexify (4c) according to (5)
6          end
7          for all the violated constraints $h_j(x_t^k) > 0$ do
8              convexify (4c) according to (5) with the closest
                 backward feasible state $x_{\hat{t}}^k$ as the new reference
                 point $x_0$, i.e. $\hat{t} = \arg\min_{\hat{t}}(h_i(x_{\hat{t}}^k) > 0)$, $\hat{t} < t$
9          end
10         solve $u^*$, $J^*(u^*)$, and the new $\{x_0^{k+1}, \ldots, x_T^{k+1}\}$
11     end
12 end
```

Figure 1 shows the convex search space about the violated reference point x_1 for a semi-convex obstacle. After including the constraints about x_1 and convexifying those via the search space of the closest feasible backward point x_0, we re-solve the problem and acquire the new trajectory (denoted by the red line), which is feasible at x_1, while the previous trajectory (denoted by the dotted line) is infeasible at x_1.

In the sequel we show how to solve the constrained LQR problem in Step 10 efficiently.

IV. Primal-Dual Controller Synthesis

After convexifying all the nonconvex constraint functions via (5), we reformulate (4) into a convex QCQP by convexifying (4c) with

$$\begin{aligned} f_i(x_t) &= {x_t}^\top H_i x_t + {c_{i|t}}^\top x_t + d_{i|t} \leq 0, t \in \mathcal{T}, i \in \mathcal{I}_t, \\ c_{i|t}^\top &= -\nabla h_i(x_t^k)^\top - (x_t^k)^\top H_i, \\ d_{i|t} &= -h_i(x_t^k) + \nabla h_i(x_t^k)^\top x_t^k + \frac{1}{2}(x_t^k)^\top H_i x_t^k, \end{aligned} \tag{6}$$

where $\mathcal{T}, \mathcal{I}_t$ represent the time and index set of the included constraints, respectively. x^k represents the k-th solution in Step 11 of the Algorithm 1. For the sake of brevity we omit the index k in (6). In the sequel, x^k is used as the k-th primal variable in primal-dual iteration.

A. Lagrangian Duality Formulation

To solve (4) with the convexified constraints (6), consider the Lagrangian $L(u, \lambda, \mu)$:

$$L(u, \lambda, \mu) = J(u) + \sum_{t=1}^{T-1} \sum_{i \in \mathcal{I}_t} \lambda_{i|t} f_i(x_t) + \sum_{t=1}^{T-1} {\mu_t}^\top \mathcal{G}_t(u_t), \tag{7}$$

where $\lambda_{t|i} \in \mathbb{R}_+$ and $\mu_t \in \mathbb{R}_+^s$ are the dual variables. u, λ, and μ are vectors consisting of u_t, $\lambda_{i|t}$, and μ_t, respectively. Accordingly, the dual function $D(\lambda, \mu)$ is defined as

$$D(\lambda, \mu) = \inf_u L(u, \lambda, \mu). \tag{8}$$

Then, from the duality theory [23], we have that

$$\sup_{\lambda \geq 0, \mu \geq 0} D(\lambda, \mu) \leq \inf_u J(u). \tag{9}$$

We now present the optimality condition of leveraging the Lagrangian formulation to solve the constrained LQR problem directly derived from the KKT conditions and strong convexity propositions.

Proposition 1. *Suppose Slater's condition holds, i.e. there exists a series of inputs $\tilde{u}$ such that $x_t^\top H_i x_t + c_{i|t}^\top x_t + d_{i|t} < 0$ and $G_t \tilde{u}_t + e_t < 0$. Then there exists an optimal control law $u_t^*(x_t, \lambda^*, \mu^*)$ associated with dual variables λ^*, μ^*, which are defined as the maximum of Lagrangian $D(\lambda, \mu)$:*

$$\lambda^*, \mu^* = \arg\max_{\lambda \geq 0, \mu \geq 0} D(\lambda, \mu).$$

Moreover, the following conditions hold:

1) *Duality gap is zero:*
   $L(u^*, \lambda^*, \mu^*) = \min_u L(u, \lambda^*, \mu^*) = D(\lambda^*, \mu^*)$*;*
2) *Inequality constraints* (4d) *and* (6) *hold;*
3) *Complementary slackness holds:*
   $\lambda_{i|t}^* f_i(x_t^*) = 0$*,* $\forall t \in \mathcal{T}, i \in \mathcal{I}_t$*;*
   $\mu_t^* G_t(u_t^*) = 0$*,* $t = 0, \ldots, T-1$*.*

Proof. 1) follows from the standard duality theorem, while 2) and 3) come from KKT conditions. □

B. Primal-Dual Approach

A primal-dual approach is used to solve the constrained LQR problem. The dual function $D(\lambda, \mu)$ is concave in λ and μ The gradient expressions $\nabla D(\lambda_{i|t})$ and $\nabla D(\mu_t)$ are:

$$\begin{aligned} \nabla D(\lambda_{i|t}) &= {x_t}^\top H_i x_t + {c_{i|t}}^\top x_t + d_{i|t}, \\ \nabla D(\mu_t) &= G_t u_t + e_t. \end{aligned} \tag{10}$$

Algorithm 2 shows the primal-dual approach for solving the constrained LQR. The step sizes α^k fulfill that i) $\sum\limits_{k \to \infty} \alpha^k \to \infty$; ii) $\sum\limits_{k \to \infty} \left(\alpha^k\right)^2 < \infty$. Then through the convergence results of the dual ascent method, $[u^k, \lambda^k, \mu^k]$ converge to the saddle point of $L(u, \lambda, \mu)$ with sublinear convergence rate $\mathcal{O}(\frac{1}{\sqrt{k}})$ [23].

The quadratically convexified constraints enable us to solve the optimisation sub-problem in Step 2. An explicit minimum can be derived through dynamic programming with every specific dual variable, since there is no additional constraints other than system dynamics constraints in the sub-optimisation problem.

C. Optimal Safety-Critical Control Laws

The solution to the optimisation sub-problem in step 2 of Algorithm 2 is shown in this subsection. The analysis uses the Hamilton-Jacobi-Bellman (HJB) equation and the Pontryagin Minimum Principle. We first define the auxiliary quadratic cost matrix

$$Q_{\lambda|t} \triangleq Q + \sum_{i \in \mathcal{I}_t} \lambda_{i|t} H_i. \tag{11}$$

where $Q_{\lambda|t} \succeq 0$ since $Q \succeq 0$ and $H_i \succeq 0$.

For simplicity of notation in the sequel, the following substitutions are used

$$\Lambda_t = \begin{bmatrix} \lambda_{1|t} & & \cdots & \lambda_{\mathcal{I}_t|t} & \\ & \lambda_{1|t} & \cdots & & \lambda_{\mathcal{I}_t|t} \end{bmatrix}^\top,$$

$$C_t = [c_{1|t}, \ldots, c_{\mathcal{I}_t|t}]^\top, d_t = [d_{1|t}, \ldots, d_{\mathcal{I}_t|t}]^\top,$$

where $\lambda_t = [\lambda_{1|t}, \ldots, \lambda_{\mathcal{I}_t|t}]^\top$. $J(u)$ in (4a) is replaced by $J(u, \lambda, \mu)$ with (6):

$$J(u, \lambda, \mu) = g_T(x, \lambda) + \sum_{t=0}^{T-1} g_t(x, u, \lambda, \mu), \tag{12}$$

where the terminal term g_T and interval term g_t are defined from (7):

$$\begin{aligned} g_T(x) &= {x_T}^\top P x_T, \\ g_t(x, u, \lambda, \mu) &= {x_t}^\top Q_{\lambda|t} {x_t}^\top + C_t^\top \Lambda_t x_t + {\lambda_t}^\top d_t \\ &\quad + \mu_t^\top (G_t u_t + e_t) + {u_t}^\top R u_t. \end{aligned} \tag{13}$$

The quadratically convexified constraints enables us to exploit closed-form expressions for both minimum and optimal control law. Let ${V_t}^*(x, \lambda, \mu)$ denote the optimal cost-to-go with dual variables λ, μ at time t:

$${V_t}^*(x, \lambda, \mu) \triangleq \min_u g_T(x) + \sum_{k=t}^{T-1} g_k(x, u, \lambda, \mu). \tag{14}$$

We note here the control admissible set $\mathcal{U}_t$ and the state admissible set $\mathcal{X}$ are not included here, since these constraints have been lifted into the objective function. The minimum is always attained since $J(u, \lambda, \mu)$ is convex over u. An explicit expression of cost-to-go function ${V_t}^*(x, \lambda, \mu)$, and optimal control law ${u_t}^*(x, \lambda, \mu)$ via dynamic programming with fixed dual variable λ and μ is given in Theorem 1.

Theorem 1. *With fixed dual variables λ, μ, for $t \leq T-1$ the closed form expression of the optimal cost-to-go function ${V_t}^*(x, \lambda, \mu)$ is expressed as:*

$${V_t}^*(x, \lambda, \mu) = {x_t}^\top F_t x_t + {S_t}^\top x_t + r_t. \tag{15}$$

Moreover, the optimal control law associated with the dual variables λ, μ is given by:

$$u_t^*(x, \lambda, \mu) = -k_t x + l_t. \tag{16}$$

Algorithm 2: Primal-dual approach for convex constrained LQR

```
Input: initial multiplier $\lambda^0 \geq 0, \mu^0 \geq 0$, a series of
       step-sizes $\alpha^k$, tolerance $\epsilon$
Output: multiplier $\lambda^k, \mu^k$
1 while $||J(u^{k+1}) - J(u^k)|| > \epsilon$ do
2     solve $u^{k+1} = \arg\min_u L(u, \lambda^k, \mu^k)$, s.t. (1)
3     update the multiplier $\lambda_{i|t}^{k+1}$ and $\mu_t^{k+1}$ through the
        gradient in (10): $\lambda_{i|t}^{k+1} = [\lambda^{k+1} + \alpha^k \nabla D(\lambda_{i|t}^k)]_+$,
        $\mu_t^{k+1} = [\mu^k + \alpha^k \nabla D(\mu_t^k)]_+$ for each $t \in \mathcal{T}, i \in \mathcal{I}_t$
4 end
```

The discrete time backward recursions through dynamic programming are given by:

$$\begin{aligned} F_{t-1} &= -A^\top F_t B M_{t-1}^{-1} B^\top F_t A + Q_{\lambda|t} + A^\top F_t A, \\ S_{t-1}^\top &= C_{t-1}^\top \Lambda_{t-1} + S_t^\top A \\ &\quad - (S_t^\top B + \mu_{t-1} G_{t-1}) M^{-1} B^\top F_t{}^\top A, \\ r_{t-1} &= \lambda_{t-1}{}^\top d_{t-1} - \mu_{t-1} e_{t-1} + r_t, \\ M_{t-1} &= B^\top F_t B + R, \\ k_{t-1} &= M_{t-1}^{-1} B^\top F_t^\top A, \\ l_{t-1} &= M_{t-1}^{-1}(B^\top S_t + G_{t-1}^\top \mu_{t-1}), \end{aligned} \tag{17}$$

with terminal conditions $F_T = P$, $S_T = 0$, $r_T = 0$.

Proof. Since $Q_\lambda - Q \succeq 0$, and $(A, \sqrt{Q})$ is detectable, the pair $(A, \sqrt{Q_\lambda})$ is also detectable. Under the assumption that (A, B) is stabilizable, the optimal cost-to-go is finite and the optimal control law can stabilize the system. We can then assume that the optimal cost-to-go function $V_t^*(x, \lambda, \mu)$ takes the quadratic form $x_t^\top F_t x_t + S_t^\top x_t + r_t$. By the HJB equation for a finite time objective we have:

$$\begin{aligned} & x_t{}^\top F_t x_t + S_t{}^\top x_t + r_t = \\ & \min_u [x_{t-1}{}^\top F_t x_{t-1} + S_{t-1}{}^\top x_{t-1} + r_{t-1} + g_t(x, u, \lambda, \mu)]. \end{aligned} \tag{18}$$

Setting the derivative of u over t to zero yields (16). Substituting the optimal control law in (16) for u_t in (18), and noticing that the quadratic, linear and constant terms are the same for both side of the equation, results in (17). □

Theorem 1 presents the discrete recursive law for the constrained LQR with fixed λ and μ. The method for fixing λ and μ is shown in Subsection IV-B.

Proposition 2. *Suppose that all the constraints are inactive within time interval $[k, T]$, then the optimal control law simplifies into $u_t^*(x, 0, 0) = -(R + B^\top F_{t+1} B)^{-1} B^\top F_{t+1} A x_t$, where F_{t+1} is the solution of the algebraic Riccati equation $\dot{P} + PA + A^\top P - PBR^{-1}B^\top P - Q = 0$ at time $t+1$.*

Proof. When all the constraints are inactive, the states $\{x_k, \ldots, x_T\}$ satisfy the inequality constraints strictly. Therefore, from Proposition 1, the corresponding dual variables satisfy $\{\lambda_k = 0, \mu_k = 0, \ldots, \lambda_T = 0, \mu_T = 0\}$. We then have $Q_{\lambda|t} = Q, \forall t \in [k, T]$ in (17), which immediately implies that the dynamics of F_t are given by the standard algebraic Riccati equation. Hence, $S_T^\top = 0, S_{T-1}^\top = C_{t-1}^\top \Lambda_{t-1} + S_t^\top A - (S_t^\top B + \mu_{t-1} G_{t-1}) M^{-1} B^\top F_t{}^\top A = 0, \ldots, S_k^\top = 0$. With the recursive law we can see that $S_t = 0, \forall t = k, \ldots, T$. □

Proposition 2 gives a theoretical illustration of what happens when the state enters the *invariant set*: in this case, the residual problem can be solved by means of unconstrained LQR. This is the backbone of solving the infinite horizon constrained LQR. We then immediately prove that for any $\lambda, \mu \geq 0$, the optimal law (16) stabilizes the system.

Theorem 2. *For the given fixed dual variables λ, μ, the control law $u_t^*(x_t, \lambda, \mu)$ stabilizes the system.*

Proof. Since $Q_\lambda \succeq Q$ and $(A, \sqrt{Q})$ is detectable, the pair $(A, \sqrt{Q_\lambda})$ is also detectable. The feedback quantity k_t of the control law $u_t^*(x, \lambda)$ is $-(R + B^\top F_{t+1} B)^{-1} B^\top F_{t+1} A$. This implies that the spectral radius $\rho(A + BK) < 0$, which proves asymptotic stability of the close-loop system. □

V. Simulation & Experiment

In this section we verify the proposed method through: i) the effectiveness of the algorithm in terms of the trajectory's feasibility and stability of the equilibrium; ii) the computation time compared to that of nonlinear solvers i.e. interior-point (Ipopt), SQP, SQP-legacy (SQP-L), active-set (act-set), and incremental SCA (iSCA) [13]; iii) feasibility comparison with the above solvers and iSCA.

We consider three sets of experiments. In the first and second sets, we consider scenarios with 5 and 15 random irregular shaped semi-convex obstacles along with a 100-length planning horizon for numerical verification. In the third set we test our method with an Omnidirectional robot on the testbed [24] to verify the effectiveness in practice. The testbed involves 7 irregular shaped obstacles. The goal of the robot is to reach the endpoint stably without colliding with the obstacle, as well as minimizing the cost.

All numerical experiments are performed in MATLAB on a computer with an Intel(R) Core(TM) i9-9980XE CPU, 3.00GHz processor and 64GB RAM. The hardware implementation on the testbed is performed with the Robopheus reality testbed [24].

A. Numerical Simulation

In the first scenario, the workspace is a 4×4 square with 5 irregular shaped obstacles randomly placed inside. The obstacle coverage rate is 44.3% at the central area (from [1.15, 0.3] to [3.36, 3.6]). The tolerance ϵ in Algorithm 2 is $\epsilon = 0.7$. The length of planning horizon is set to 100. The start point is $[4, 3.6]$, and the desired endpoint is $[0, 0]$. The control input is bounded by a box constraint $[-0.7, 0.7]$ on both x and y directions. The trajectory and speed calculated by the proposed method are shown in Fig. 2.

The active-set solver is the only method that produced a collision-free trajectory among the nonlinear solvers used. Our method also results in a collision-free trajectory, with a much smaller cost (96.02 compared to 283.96 of the trajectory returned by the active-set solver). The iSCA is infeasible at some points. Details of infeasibility are shown in the magnification,

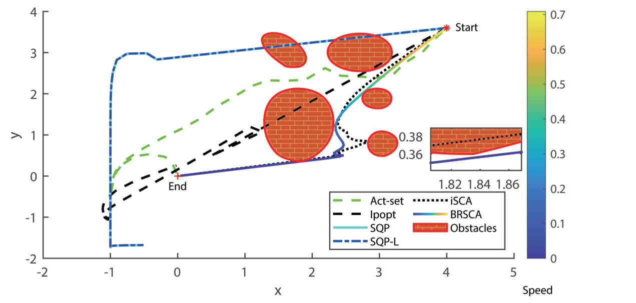

Fig. 2: Comparison between BRSCA and nonlinear solvers, iSCA with 5 obstacles in numerical simulation

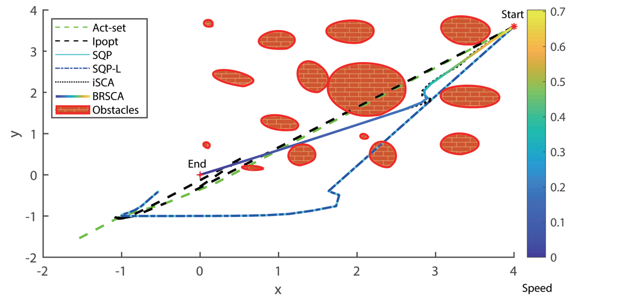

Fig. 3: Comparsion between BRSCA and nonlinear solvers, iSCA with 15 obstacles in numberical simulation

where the dot, red, and blue line denote the trajectory of iSCA, the boundary of the obstacle and the trajectory of BRSCA, respectively. As discussed before, infeasibility is caused by possible linearization about included points that are infeasible.

In the second scenario, all the settings are the same to those of the first scenario, except that the number of obstacles is 15. As shown in Fig. 3, BRSCA provided a feasible trajectory and stable control for obstacle avoidance when encountering with more constraints. The iSCA is the only other method that resulted in a collision-free trajectory. The cost value of iSCA is 98.4897, which is slightly bigger than 96.7878 of ours. The other methods did not perform well when 1700 constraints are included (the trajectory is infeasible).

B. Hardware Implementation

We validate BRSCA on a hardware testbed, which is a 5m $\times$ 3m rectangular space with 7 irregular obstacles randomly placed inside. The obstacle coverage rate is about 9.3%. The tolerance in Algorithm 2 is set to $\epsilon = 0.1$, and the length of planning horizon in this scenario is 200. The start point is randomly set at $[429, 207]$, and the endpoint is $[38, 0]$ but not the origin in this case to avoid edge distortion of the cameras on the boundary. The control frequency is 25Hz. Fig. 4 depicts the collision-free trajectory, as well as the velocity variations. Stability around the endpoint is demonstrated by the diminished velocity.

TABLE I: Computation times of different methods

Obs	BRSCA	iSCA*	Ipopt*	SQP*	SQP-L*	Act-set*
5	0.74	1.30	9.88	9.17	10.67	163.89
7	0.98	1.24	9.68	31.38	38.21	70.28
9	1.04	1.26	10.47	91.24	124.13	70.58
12	1.23	1.31	11.83	122.22	174.16	76.67
15	1.36	1.37	13.49	27.91	38.48	220.48
Time**	1.00	1.21	10.35	52.71	72.10	112.54

* The method cannot guarantee the feasibility of the results.
** The average time compared to BRSCA.

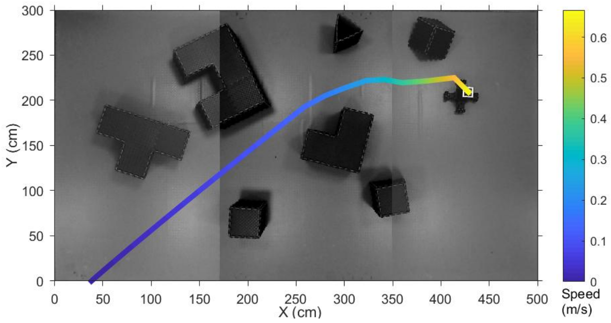

Fig. 4: Experiment with 7 obstacles

C. Computation Time

Table I shows a comparison of the computation time between BRSCA and nonlinear solvers, iSCA. It can be seen that BRSCA can solve the problem with multiple obstacles within around 1 second. Besides, the computation speed grows linearly with the number of obstacles empirically. The observation that the computation speed grows moderately with the number of obstacles or even decreases for other algorithms is due to the fact that they fail in finding feasible solutions, e.g. SQP with 15 obstacles. The performance of BRSCA is illustrated better, even though the comparison is somewhat unfair for ours. In summary, BRSCA has higher probability of finding feasible solutions and reduces the computation time by at least 17.4%.

TABLE II: Computation time with different tolerance settings and numbers of obstacles

ϵ	5 obs	7 obs	9 obs	12 obs	15 obs
0.0003	58.78	70.15	80.71	96.78	114.31
0.03	31.89	38.58	44.86	54.39	64.66
0.7	0.74	0.98	1.04	1.23	1.36

Table II shows the computation time of BRSCA with different tolerance settings and numbers of obstacles. With a modest tolerance such as 0.7 BRSCA is real-time implementable. We note here with an appropriate stepsize selection in Algorithm 2 can realize further acceleration. Selections of stepsize and an accelerated primal-dual approach will be investigated in our future work.

VI. Conclusion

This paper develops the BRSCA algorithm for motion planning by means of linear quadratic regulator formulation. We demonstrate the relative merits of our approach with incremental SCA, which uses a similar principle. We also explore the special structure of the formulated constrained LQR problem by transforming it into a convex QCQP. We prove that the Lagrangian dual problem can be regarded as an unconstrained LQR for every dual variable. The unconstrained LQR is proved to have stability guarantees for every dual variable, thereby proving the stability during primal-dual iterations. For every dual variable we use backward recursion to give closed-form solutions for the optimal cost-to-go and feedback control law. The proposed algorithm is validated on a hardware testbed and by means of numerical simulations. In our future work, we aim at developing a decentralized algorithm for multi-robot systems.

References

[1] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014.

[2] J. Yao, C. Lin, X. Xie, A. J. Wang, and C.-C. Hung, "Path planning for virtual human motion using improved a* star algorithm," in *2010 Seventh international conference on information technology: new generations*, pp. 1154–1158, IEEE, 2010.

[3] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[4] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic foundations of robotics XI*, pp. 109–124, Springer, 2015.

[5] R. Deits and R. Tedrake, "Efficient mixed-integer planning for uavs in cluttered environments," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 42–49, IEEE, 2015.

[6] J. Tordesillas, B. T. Lopez, and J. P. How, "Faster: Fast and safe trajectory planner for flights in unknown environments," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 1934–1940, IEEE, 2019.

[7] J. Tordesillas and J. P. How, "Mader: Trajectory planner in multiagent and dynamic environments," *IEEE Transactions on Robotics*, 2021.

[8] H. A. Le Thi and T. P. Dinh, "Dc programming and dca: thirty years of developments," *Mathematical Programming*, vol. 169, no. 1, pp. 5–68, 2018.

[9] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optimization and Engineering*, vol. 17, no. 2, pp. 263–287, 2016.

[10] C. Liu, C.-Y. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM Journal on Control and optimization*, vol. 56, no. 4, pp. 2712–2733, 2018.

[11] Y. Mao, D. Dueri, M. Szmuk, and B. Açıkmeşe, "Successive convexification of non-convex optimal control problems with state constraints," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4063–4069, 2017.

[12] M. Szmuk, T. P. Reynolds, and B. Açıkmeşe, "Successive convexification for real-time six-degree-of-freedom powered descent guidance with state-triggered constraints," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 8, pp. 1399–1413, 2020.

[13] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5954–5961, IEEE, 2015.

[14] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *2012 IEEE/RSJ international conference on Intelligent Robots and Systems*, pp. 1917–1922, IEEE, 2012.

[15] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[16] P. O. Scokaert and J. B. Rawlings, "Constrained linear quadratic regulation," *IEEE Transactions on automatic control*, vol. 43, no. 8, pp. 1163–1169, 1998.

[17] L. Ferranti, G. Stathopoulos, C. N. Jones, and T. Keviczky, "Constrained lqr using online decomposition techniques," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 2339–2344, IEEE, 2016.

[18] Y. Chen, M. Ahmadi, and A. D. Ames, "Optimal safe controller synthesis: A density function approach," in *2020 American Control Conference (ACC)*, pp. 5407–5412, IEEE, 2020.

[19] G. Stathopoulos, M. Korda, and C. N. Jones, "Solving the infinite-horizon constrained lqr problem using accelerated dual proximal methods," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1752–1767, 2016.

[20] Y. Aoyama, G. Boutselis, A. Patel, and E. A. Theodorou, "Constrained differential dynamic programming revisited," *arXiv preprint arXiv:2005.00985*, 2020.

[21] B. Plancher, Z. Manchester, and S. Kuindersma, "Constrained unscented dynamic programming," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5674–5680, IEEE, 2017.

[22] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in *International Workshop on Hybrid Systems: Computation and Control*, pp. 477–492, Springer, 2004.

[23] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[24] X. Ding, H. Wang, H. Li, H. Jiang, and J. He, "Robopheus: A virtual-physical interactive mobile robotic testbed," *arXiv preprint arXiv:2103.04391*, 2021.