SIAM J. CONTROL OPTIM.
Vol. 56, No. 4, pp. 2712–2733




# THE CONVEX FEASIBLE SET ALGORITHM FOR REAL TIME OPTIMIZATION IN MOTION PLANNING*

CHANGLIU LIU†, CHUNG-YEN LIN†, AND MASAYOSHI TOMIZUKA†

**Abstract.** With the development of robotics, there are growing needs for real time motion planning. However, due to obstacles in the environment, the planning problem is highly nonconvex, which makes it difficult to achieve real time computation using existing nonconvex optimization algorithms. This paper introduces the convex feasible set algorithm, which is a fast algorithm for nonconvex optimization problems that have convex costs and nonconvex constraints. The idea is to find a convex feasible set for the original problem and iteratively solve a sequence of subproblems using the convex constraints. The feasibility and the convergence of the proposed algorithm are proved in the paper. The application of this method on motion planning for mobile robots is discussed. The simulations demonstrate the effectiveness of the proposed algorithm.

**Key words.** nonconvex optimization, nondifferentiable optimization, robot motion planning

**AMS subject classifications.** 90C55, 90C26, 68T40, 93C85

**DOI.** 10.1137/16M1091460

## 1. Introduction.

Although great progress has been made in robot motion planning [13], the field is still open for research regarding real time planning in a dynamic uncertain environment. The applications include but are not limited to real time navigation [8], autonomous driving [12, 16], robot arm manipulation, and human robot cooperation [15]. To achieve safety and efficiency, robot motion should be replanned from time to time when new information is obtained during operation. The motion planning algorithm should run fast enough to meet the real time requirement for planning and replanning.

In this paper, focus on optimization-based motion planning methods, which fit into the framework of model-predictive control [10], where the optimal trajectory is obtained by solving a constrained optimization at each time step. As there are obstacles in the environment, the constraints are highly nonconvex, which makes the problem hard to solve in real time. Various methods have been developed to deal with the nonconvexity [22, 27]. One popular way is through convexification [28], e.g., transforming the nonconvex problem into a convex one. Some authors propose to transform the nonconvex problem to semidefinite programming [6]. Some authors introduce lossless convexification by augmenting the space [1, 9]. And some authors use successive linear approximation to remove nonconvex constraints [18, 19]. However, the first method can only handle quadratic cost functions. The second approach highly depends on the linearity of the system and may not be able to handle diverse obstacles. The third approach may not generalize to nondifferentiable problems. Another widely used convexification method is the sequential quadratic programming (SQP) [26, 29], which approximates the nonconvex problem as a sequence of quadratic programming problems and solves them iteratively. The method has been successfully applied to offline robot motion planning [11, 25]. However, as SQP is a generic algorithm, the

*Received by the editors August 29, 2016; accepted for publication (in revised form) May 17, 2018; published electronically July 19, 2018.
http://www.siam.org/journals/sicon/56-4/M109146.html
**Funding:** The research was supported by FANUC Corporation.
†Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA 94720 (changliuliu@berkeley.edu, chung_yen@berkeley.edu, tomizuka@berkeley.edu).

unique geometric structure of the motion planning problems is neglected, which usually results in failure to meet the real time requirement in engineering applications.

Typically, in a motion planning problem, the constraints are physically determined, while the objective function is designed to be convex [31, 23]. The nonconvexity mainly comes from the physical constraints. Regarding this observation, a fast algorithm called the convex feasible set (CFS) algorithm is proposed in this paper to solve optimization-based motion planning problems with convex objective functions and nonconvex constraints.

The main idea of the CFS algorithm is to transform the original problem into a sequence of convex subproblems by obtaining CFSs within the nonconvex domain, then iteratively solve the convex subproblems until convergence. The idea is similar to SQP in that it tries to solve several convex subproblems iteratively. The difference between CFS and SQP lies in the way to obtain the convex subproblems. The geometric structure of the original problem is fully considered in CFS. This strategy will make the computation faster than conventional SQP and other nonconvex optimization methods such as interior point method (ITP) [32], as will be demonstrated later. Moreover, local optima is guaranteed.

It is worth noting that the convex feasible set in the trajectory space can be regarded as a convex corridor. The idea of using convex corridors to simplify the motion planning problems has been discussed in [3, 33]. However, these methods are application-specific without theoretical guarantees. In this paper, we consider general nonconvex and nondifferentiable optimization problems (which may not only arise from motion planning problems, but also other problems) and provide theoretical guarantees of the method.

The remainder of the paper is organized as follows. Section 2 proposes a benchmark optimization problem. Section 3 discusses the proposed CFS algorithm in solving the benchmark problem. Section 4 shows the feasibility and convergence of the algorithm. Section 5 illustrates the application of the algorithm on motion planning problems for mobile robots. Section 6 concludes the paper.

## 2. The optimization problem.

### 2.1. The benchmark problem.

Consider an optimization problem with a convex cost function but nonconvex constraints, i.e.,

$$\min_{\mathbf{x}\in\Gamma} J(\mathbf{x}), \tag{1}$$

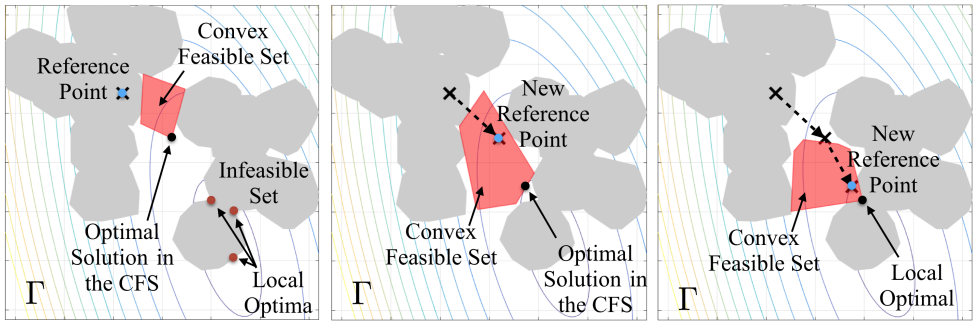
where $\mathbf{x} \in \mathbb{R}^n$ is the decision variable and the problem follows two assumptions.

*Assumption* 1 (cost). $J : \mathbb{R}^n \to \mathbb{R}^+$ is smooth and strictly convex.

*Assumption* 2 (constraint). The set $\Gamma \subset \mathbb{R}^n$ is connected and closed, with piecewise smooth and non-self-intersecting boundary $\partial\Gamma$. For every point $x \in \Gamma$, there exists an $n$-dimensional convex polytope $P \subset \Gamma$ such that $x \in P$.

Assumption 1 implies that $J$ is radially unbounded, i.e., $J(\mathbf{x}) \to \infty$ when $\|\mathbf{x}\| \to \infty$. Assumption 2 specifies the geometric features of the feasible set $\Gamma$, where the first part deals with the topological features of $\Gamma$ and the second part ensures that there is a convex neighborhood for any point in $\Gamma$. Note that equality constraints in $\Gamma$ are excluded by Assumption 2 as the dimension of the neighborhood for any point satisfying an equality constraint is strictly less than $n$.

The geometric structure of problem (1) is in Figure 1(a). The contour represents the cost function $J$, while the gray parts represent $\Gamma^c$. There are two disjoint




(a) Iteration 1. (b) Iteration 2. (c) Iteration 3.

FIG. 1. *Geometry of problem* (1) *and the idea of the CFS algorithm.*

components in $\Gamma^c$. The goal is to find a local optimum (hopefully global optimum) starting from the initial reference point (blue dot). As shown in Figure 1(a), the problem is highly nonconvex and the nonconvexity comes from the constraints. To make the computation efficient, we propose the convex feasible set algorithm in this paper, which transforms problem (1) into a sequence of convex optimizations by obtaining a sequence of convex feasible sets inside the nonconvex domain $\Gamma$. As shown in Figure 1, the idea is implemented iteratively. At the current iteration, a convex feasible set for the current reference point (blue dot) is obtained. The optimal solution in the convex feasible set (black dot) is set as the reference point for the next iteration. The formal mathematical description of this algorithm will be discussed in section 3. The feasibility of this method, i.e., the existence of an $n$-dimensional convex feasible set, is implied by Assumption 2. Nonetheless, in order to compute the convex feasible set efficiently, we still need an analytical description of the constraint, which will be discussed in subsection 2.2.

**2.2. Analytical representation of the constraints.** The set $\Gamma$ will be represented analytically by several inequality constraints. It is called a semiconvex decomposition of $\Gamma$ if

$$\Gamma = \bigcap_{i=1}^{N} \{\mathbf{x} : \phi_i(\mathbf{x}) \geq 0\} = \bigcap_{i=1}^{N} \Gamma_i \tag{2}$$

for $N$ continuous, piecewise smooth, and semiconvex functions $\phi_i : \mathbb{R}^n \to \mathbb{R}$ such that $\Gamma_i := \{\mathbf{x} : \phi_i(\mathbf{x}) \geq 0\}$ with $\partial\Gamma_i = \{\mathbf{x} : \phi_i(\mathbf{x}) = 0\}$. Semiconvexity [5] of $\phi_i$ implies that there exists a positive semidefinite $H_i^* \in \mathbb{R}^{n\times n}$ such that the function

$$\tilde{\phi}_i(\mathbf{x}) := \phi_i(\mathbf{x}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_0)^T H_i^* (\mathbf{x} - \mathbf{x}_0) \tag{3}$$

is convex in $\mathbf{x}$ for any $\mathbf{x}_0 \in \mathbb{R}^n$. Or in other words, the hessian of $\phi_i$ is bounded below. Note that $\Gamma_i^c$'s are not required to be disjoint and $N$ should be greater than or equal to the number of disjoint components in $\Gamma^c$. The decomposition from $\Gamma$ to $\Gamma_i$'s is not unique. Neither is the function $\phi_i$ that represents $\Gamma_i$. In many cases, $\phi_i$ can be chosen as a signed distance function to $\partial\Gamma_i$, which will be discussed in subsection 5.2.

Before introducing more conditions on the decomposition (2), analytical properties of the functions $\phi_i$'s will be studied first. Since $\tilde{\phi}_i$ is convex, then for any $\mathbf{x}_0, v \in \mathbb{R}^n$, $\tilde{\phi}_i(\mathbf{x}_0 + v) - 2\tilde{\phi}_i(\mathbf{x}_0) + \tilde{\phi}_i(\mathbf{x}_0 - v) \geq 0$. Considering (3), the following

inequality holds for any semiconvex functions:

$$\phi_i(\mathbf{x}_0 + v) - 2\phi_i(\mathbf{x}_0) + \phi_i(\mathbf{x}_0 - v) \geq -v^T H_i^* v. \tag{4}$$

Moreover, since convex functions are locally Lipschitz [30], $\phi_i$ is also locally Lipschitz as implied by definition (3). However, as $\phi_i$ is only piecewise smooth, it may not be differentiable everywhere. For any $v \in \mathbb{R}^n$, define the one-side directional derivative $\partial_v$[1] as

$$\partial_v \phi_i(\mathbf{x}) := \lim_{a \to 0^+} \frac{\phi_i(\mathbf{x} + av) - \phi_i(\mathbf{x})}{a}. \tag{5}$$

For any $\|v\| = 1$, $\partial_v \phi_i(\mathbf{x})$ is bounded locally since $\phi_i$ is locally Lipschitz. If $\phi_i$ is smooth at direction $v$ at point $\mathbf{x}$, then

$$\lim_{a \to 0^+} \frac{\phi_i(\mathbf{x} + av) - \phi_i(\mathbf{x})}{a} = \lim_{a \to 0^-} \frac{\phi_i(\mathbf{x} + av) - \phi_i(\mathbf{x})}{a} = \lim_{a \to 0^+} \frac{\phi_i(\mathbf{x} - av) - \phi_i(\mathbf{x})}{-a}, \tag{6}$$

where the second equality is by taking the negative of $a$. By definition (5), the right-hand side of (6) equals to $-\partial_{-v}\phi_i(\mathbf{x})$, which implies that $\partial_v \phi_i(\mathbf{x}) = -\partial_{-v}\phi_i(\mathbf{x})$. Let $\mathcal{S}(\phi_i, \mathbf{x}) := \{v \in \mathbb{R}^n : \partial_v \phi_i(\mathbf{x}) + \partial_{-v}\phi_i(\mathbf{x}) = 0\}$ denote all the smooth directions of function $\phi_i$ at point $\mathbf{x}$. The directional derivatives satisfy the following properties.

LEMMA 1 (properties of directional derivatives). *If $\phi_i$ is continuous, piecewise smooth, and semiconvex, then for any $\mathbf{x}, v, v_1, v_2 \in \mathbb{R}^n$, and $b \in \mathbb{R}$ such that $v = v_1 + v_2$, the following inequalities hold:*

$$0 \leq \partial_v \phi_i(\mathbf{x}) + \partial_{-v}\phi_i(\mathbf{x}), \tag{7}$$

$$b\partial_v \phi_i(\mathbf{x}) \leq \partial_{bv}\phi_i(\mathbf{x}), \tag{8}$$

$$\partial_v \phi_i(\mathbf{x}) \leq \partial_{v_1}\phi_i(\mathbf{x}) + \partial_{v_2}\phi_i(\mathbf{x}). \tag{9}$$

*The equalities in* (7) *and* (8) *are achieved when $v \in \mathcal{S}(\phi_i, \mathbf{x})$. The equality in* (9) *is achieved when $v_1, v_2 \in \mathcal{S}(\phi_i, \mathbf{x})$.*

*Proof.* If $\phi_i$ is semiconvex, (4) implies that for any scalar $a$ and vector $v$,

$$\frac{\phi_i(\mathbf{x} + av) - 2\phi_i(\mathbf{x}) + \phi_i(\mathbf{x} - av)}{a} \geq -av^T H_i^* v.$$

Let $a \to 0^+$. The left-hand side approaches $\partial_v \phi_i(\mathbf{x}) + \partial_{-v}\phi_i(\mathbf{x})$, while the right-hand side approaches 0 in the limits. Hence (7) holds. By definition (5), $\partial_{bv}\phi_i(\mathbf{x}) = b\partial_v \phi_i(\mathbf{x})$ when $b \geq 0$. When $b < 0$, by (7), $-|b|\partial_v \phi_i(\mathbf{x}) \leq |b|\partial_{-v}\phi_i(\mathbf{x}) = \partial_{-|b|v}\phi_i(\mathbf{x}) = \partial_{bv}\phi_i(\mathbf{x})$. Hence (8) holds. The equality holds when $\partial_v \phi_i(\mathbf{x}) + \partial_{-v}\phi_i(\mathbf{x}) = 0$, i.e., $v \in \mathcal{S}(\phi_i, \mathbf{x})$. Moreover, (4) also implies

$$\begin{aligned} -\frac{a^2}{4}(v_1 - v_2)^T H_i^*(v_1 - v_2) &\leq \phi_i(\mathbf{x} + av_1) - 2\phi_i\left(\mathbf{x} + a\frac{v}{2}\right) + \phi_i(\mathbf{x} + av_2) \\ &= \phi_i(\mathbf{x} + av_1) - \phi_i(\mathbf{x}) + \phi_i(\mathbf{x} + av_2) - \phi_i(\mathbf{x}) - 2[\phi_i\left(\mathbf{x} + a\frac{v}{2}\right) - \phi_i(\mathbf{x})]. \end{aligned}$$

Divide both sides by $a$ and take $a \to 0^+$. Then the left-hand side approaches 0, while

[1] Note that $\partial$ refers to a boundary when followed by a set, e.g., $\partial\Gamma$. It means derivative when followed by a function, e.g., $\partial_v \phi_i$.

the right-hand side approaches $\partial_{v_1}\phi_i(\mathbf{x})+\partial_{v_2}\phi_i(\mathbf{x})-\partial_v\phi_i(\mathbf{x})$. Hence (9) holds. When $v_1, v_2 \in \mathcal{S}(\phi_i, \mathbf{x})$,

$$0 \le \partial_v\phi_i(\mathbf{x}) + \partial_{-v}\phi_i(\mathbf{x}) \le \partial_{v_1}\phi_i(\mathbf{x}) + \partial_{v_2}\phi_i(\mathbf{x}) + \partial_{-v_1}\phi_i(\mathbf{x}) + \partial_{-v_2}\phi_i(\mathbf{x}) = 0. \tag{10}$$

The first inequality is due to (7); the second inequality is due to (9). Hence the equality in (9) is attained. ∎

Define the subdifferential of $\phi_i$ at $\mathbf{x}$ as

$$D\phi_i(\mathbf{x}) := \{d \in \mathbb{R}^n : d \cdot v \le \partial_v\phi_i(\mathbf{x}) \ \forall v \in \mathbb{R}^n\}. \tag{11}$$

The validity of the definition, i.e., that the right-hand side of (11) is nonempty, can be verified by Lemma 1. By (8) and (9), for any $v_1, v_2 \in \mathcal{S}(\phi_i, \mathbf{x})$, $a, b \in \mathbb{R}$, and $v = av_1+bv_2$, we have $\partial_v\phi_i(\mathbf{x}) = a\partial_{v_1}\phi_i(\mathbf{x})+b\partial_{v_2}\phi_i(\mathbf{x})$ and $\partial_v\phi_i(\mathbf{x})+\partial_{-v}\phi_i(\mathbf{x}) = 0$. Hence $v \in \mathcal{S}(\phi_i, \mathbf{x})$. We can conclude that (1) $\mathcal{S}(\phi_i, \mathbf{x})$ is a linear subspace of $\mathbb{R}^n$ and (2) the function induced by the directional derivative $v \mapsto \partial_v\phi_i(\mathbf{x})$ is a sublinear function[2] on $\mathbb{R}^n$ and a linear function on $\mathcal{S}(\phi_i, \mathbf{x})$. By the Hahn–Banach theorem [7], there exists a vector $d \in \mathbb{R}^n$ such that $d\cdot v = \partial_v\phi_i(\mathbf{x})$ for $v \in \mathcal{S}(\phi_i, \mathbf{x})$ and $d\cdot v \le \partial_v\phi_i(\mathbf{x})$ for $v \in \mathbb{R}^n$. Moreover, as the unit directional derivative is bounded, the subgradients are also bounded. Hence the definition in (11) is justified. The elements in $D\phi_i(\mathbf{x})$ are called subgradients. When $\phi_i$ is smooth at $\mathbf{x}$, $D\phi_i(\mathbf{x})$ reduces to a singleton set which contains only the gradient $\nabla\phi_i(\mathbf{x})$ such that $\nabla\phi_i(\mathbf{x})\cdot v = \partial_v\phi_i(\mathbf{x})$ for all $v \in \mathbb{R}^n$. The definition (11) follows from Clarke (generalized) subgradients for nonconvex functions [4].

With the definition of subdifferential for continuous, piecewise smooth, and semiconvex functions in (11), the following assumption regarding the analytical representation is made.

*Assumption* 3 (analytical representations). For $\Gamma$ satisfying Assumption 2, there exists a semiconvex decomposition (2) such that (1) $D\phi_i(\mathbf{x}) \ne \{0\}$ for all $\mathbf{x}$, (2) $0 \notin D\phi_i(\mathbf{x})$ if $\mathbf{x} \in \partial\Gamma_i$, and (3) for any $\mathbf{x}$ such that $I := \{i : \phi_i(\mathbf{x}) = 0\} \ne \emptyset$, there exists $v \in \mathbb{R}^n$ such that $\partial_v\phi_i(\mathbf{x}) < 0$ for all $i \in I$.

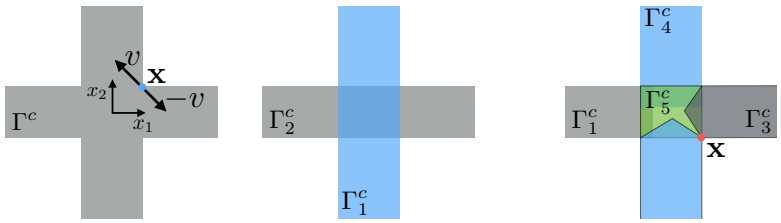
Note that the hypothesis that any $\Gamma$ that satisfies Assumption 2 has a semiconvex decomposition that satisfies Assumption 3 will be verified in our future work. A method to construct the desired $\phi_i$'s is discussed in [14].

The first condition in Assumption 3 ensures that $\phi_i$ will not have smooth extreme points. Geometrically, the second condition in Assumption 3 implies that there cannot be any concave corners[3] in $\Gamma_i^c$ or convex corners in $\Gamma_i$. Suppose $\Gamma_i^c$ has a concave corner at $\mathbf{x} \in \partial\Gamma_i$. Since $0 \notin D\phi_i(\mathbf{x})$, we can choose a unit vector $v$ such that $\partial_v\phi_i(\mathbf{x}) < 0$ and $\partial_{-v}\phi_i(\mathbf{x}) < 0$ as shown in Figure 2(a). Then (7) is violated, which contradicts the assumption on semiconvexity. Nonetheless, concave corners are allowed in $\Gamma^c$, but should only be formulated by a union of several intersecting $\Gamma_i^c$'s as shown in Figure 2(b). In the example, the set $\Gamma = \{\mathbf{x} = (x_1, x_2) : \min(|x_1| - 1, |x_2| - 1) \ge 0\}$ is partitioned into two sets $\Gamma_1 = \{\mathbf{x} : |x_1| - 1 \ge 0\}$ and $\Gamma_2 = \{\mathbf{x} : |x_2| - 1 \ge 0\}$. Both $\phi_1 = |x_1| - 1$ and $\phi_2 = |x_1| - 1$ satisfy Assumption 3. Without the partition, $\phi = \min(|x_1|-1, |x_2|-1)$ violates the condition on semiconvexity.[4] The third condition in Assumption 3 implies that once $\Gamma_i^c$'s intersect, they should have a common interior

[2] A function $f$ is called sublinear if it satisfies positive homogeneity $f(ax) = af(x)$ for $a > 0$, and subadditivity $f(x + y) \le f(x) + f(y)$.

[3] Some authors name convex corners as outer corners and concave corners as inner corners [21].

[4] Let $\mathbf{x} = (1, 1)$ and $v = (\cos\frac{\pi}{4}, \sin\frac{\pi}{4})$. Then $\phi_i(\mathbf{x}+av)-2\phi_i(\mathbf{x})+\phi_i(\mathbf{x}-av) = -a\cos\frac{\pi}{4}-a\sin\frac{\pi}{4} = -\sqrt{2}a$, which cannot be greater than any $-a^2v^TH_i^*v$ when $a$ is small.




(a) The constraint $\Gamma$. (b) Valid decomposition of $\Gamma$. (c) Invalid decomposition of $\Gamma$.

FIG. 2. *Representing* $\Gamma$ *using* $\Gamma_i$ *and* $\phi_i$.

among one another. For example, the decomposition in Figure 2(c) is not allowed. In this case, the obstacle is partitioned into five components. $\partial\Gamma_2^c$, $\partial\Gamma_3^c$, and $\partial\Gamma_5^c$ intersect at $\mathbf{x}$. However, there does not exist $v \in \mathbb{R}^n$ such that $\partial_v \phi_i(\mathbf{x}) < 0$ for all $i \in \{2, 3, 5\}$ as the interiors of $\Gamma_2^c$ and $\Gamma_3^c$ do not intersect. This condition is enforced in order to ensure that the computed convex feasible set is nonempty as will be discussed in Lemma 2.

In this following discussion, $\phi_i$'s and $\Gamma_i$'s are referred as the semiconvex decomposition of $\Gamma$ that satisfies Assumption 3.

**2.3. Physical interpretations.** Many motion planning problems can be formulated into (1) when $\mathbf{x}$ is regarded as the trajectory as will be discussed in section 5. The dimension of the problem $n$ is proportional to the number of sampling points on the trajectory. If continuous trajectories are considered, then $n \to \infty$ and $\mathbb{R}^n$ approaches the space of continuous functions $\mathcal{C}(\mathbb{R})$ in the limit.

In addition to motion planning problems, the proposed method deals with any problem with similar geometric properties as specified in Assumptions 1 and 2. Moreover, problems with global linear equality constraints also fit into the framework if we solve the problem in the low-dimensional linear manifold defined by the linear equality constraints. The case for nonlinear equality constraints is much trickier since convexification on nonlinear manifold is difficult in general. A relaxation method to deal with nonlinear equality constraints is discussed in [17].

## 3. Solving the optimization problem.

**3.1. The CFS algorithm.** To solve the problem (1) efficiently, we propose the CFS algorithm. As introduced in subsection 2.1, a convex feasible set $\mathcal{F}$ for the set $\Gamma$ is a convex set such that $\mathcal{F} \subset \Gamma$. $\mathcal{F}$ is not unique. We define the desired $\mathcal{F}$ in subsection 3.2. As $\Gamma$ can be covered by several (maybe infinitely many) convex feasible sets, we can efficiently search the nonconvex space $\Gamma$ for solutions by solving a sequence of convex optimizations constrained in a sequence of convex feasible sets. The idea is implemented iteratively as shown in Figure 1. At iteration $k$, given a reference point $\mathbf{x}^{(k)}$, a convex feasible set $\mathcal{F}^{(k)} := \mathcal{F}(\mathbf{x}^{(k)}) \subset \Gamma$ is computed around $\mathbf{x}^{(k)}$. Then a new reference point $\mathbf{x}^{(k+1)}$ will be obtained by solving the resulting convex optimization problem

$$\mathbf{x}^{(k+1)} = \arg \min_{\mathbf{x} \in \mathcal{F}^{(k)}} J(\mathbf{x}). \tag{12}$$

The optimal solution will be used as the reference point for the next step. The iteration will terminate if either the change in solution is small, e.g.,

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\| \leq \epsilon_1 \tag{13}$$

for some small $\epsilon_1 > 0$, or the descent in cost is small, e.g.,

$$J(\mathbf{x}^{(k)}) - J(\mathbf{x}^{(k+1)}) \leq \epsilon_2 \tag{14}$$

for some small $\epsilon_2 > 0$. We will show in section 4 that these two conditions are equivalent and both of them imply convergence. The process is summarized in Algorithm 1.

**Algorithm 1** The CFS algorithm.

```
Initialize initial guess x^(0), k := 0;
while True do
  Find a convex feasible set F^(k) ⊂ Γ for x^(k);
  Solve the convex optimization problem (12) for x^(k+1);
  if (13) or (14) is satisfied then
    Break the while loop;
  end if
  k := k + 1;
end while
return x^(k+1);
```

**3.2. Finding the convex feasible set.** Considering the semiconvex decomposition (2), we try to find a convex feasible set $\mathcal{F}_i$ for each constraint $\Gamma_i = \{\mathbf{x} : \phi_i(\mathbf{x}) \geq 0\}$.

*Case* 1: *$\phi_i$ is concave.* Then $\Gamma_i$ is convex. The convex feasible set is chosen to be itself,

$$\mathcal{F}_i = \Gamma_i. \tag{15}$$

*Case* 2: *$\phi_i$ is convex.* Then $\Gamma_i^c$ is convex. The convex feasible set $\mathcal{F}_i$ with respect to a reference point $\mathbf{x}^r \in \mathbb{R}^n$ is defined as

$$\mathcal{F}_i(\mathbf{x}^r) := \{\mathbf{x} : \phi_i(\mathbf{x}^r) + \hat{\nabla}\phi_i(\mathbf{x}^r)(\mathbf{x} - \mathbf{x}^r) \geq 0\}, \tag{16}$$

where $\hat{\nabla}\phi_i(\mathbf{x}^r) \in D\phi_i(\mathbf{x}^r)$ is a subgradient. When $\phi_i$ is smooth at $\mathbf{x}^r$, $\hat{\nabla}\phi_i(\mathbf{x}^r)$ equals to the gradient $\nabla\phi_i(\mathbf{x}^r)$. Otherwise, the subgradient is chosen according to the method discussed in subsection 3.3. Since $\phi_i$ is convex, $\phi_i(\mathbf{x}) \geq \phi_i(\mathbf{x}^r) + \partial_{\mathbf{x}-\mathbf{x}^r}\phi_i(\mathbf{x}^r) \geq \phi_i(\mathbf{x}^r) + d \cdot (\mathbf{x} - \mathbf{x}^r)$ for all $d \in D\phi_i(\mathbf{x}^r)$, where the second inequality is due to (11). Hence $\mathcal{F}_i(\mathbf{x}^r) \subset \{\mathbf{x} : \phi_i(\mathbf{x}) \geq 0\} = \Gamma_i$ for all $\mathbf{x}^r \in \mathbb{R}^n$.

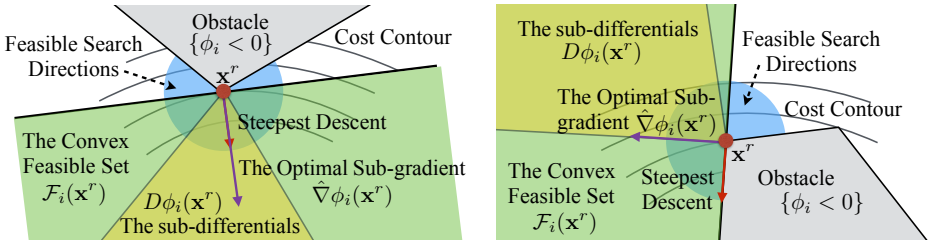
*Case* 3: *$\phi_i$ is neither concave nor convex.* Considering (3), the convex feasible set with respect to the reference point $\mathbf{x}^r$ is defined as

$$\mathcal{F}_i(\mathbf{x}^r) := \{\mathbf{x} : \phi_i(\mathbf{x}^r) + \hat{\nabla}\phi_i(\mathbf{x}^r)(\mathbf{x} - \mathbf{x}^r) \geq \frac{1}{2}(\mathbf{x} - \mathbf{x}^r)^T H_i^*(\mathbf{x} - \mathbf{x}^r)\}, \tag{17}$$

where $\hat{\nabla}\phi_i(\mathbf{x}^r) \in D\phi_i(\mathbf{x}^r)$ is chosen according to the method discussed in subsection 3.3. Since $\phi_i$ is semiconvex, $\phi_i(\mathbf{x}) \geq \phi_i(\mathbf{x}^r) + \partial_{\mathbf{x}-\mathbf{x}^r}\phi_i(\mathbf{x}^r) - \frac{1}{2}(\mathbf{x} - \mathbf{x}^r)^T H_i^*(\mathbf{x} - \mathbf{x}^r) \geq \phi_i(\mathbf{x}^r) + d \cdot (\mathbf{x} - \mathbf{x}^r) - \frac{1}{2}(\mathbf{x} - \mathbf{x}^r)^T H_i^*(\mathbf{x} - \mathbf{x}^r)$ for all $d \in D\phi_i(\mathbf{x}^r)$. Hence $\mathcal{F}_i(\mathbf{x}^r) \subset \{\mathbf{x} : \phi_i(\mathbf{x}) \geq 0\} = \Gamma_i$ for all $\mathbf{x}^r \in \mathbb{R}^n$.

Considering (15), (16), and (17), the convex feasible set for $\Gamma$ at $\mathbf{x}^r$ is defined as

$$\mathcal{F}(\mathbf{x}^r) := \bigcap_{i=1}^{N} \mathcal{F}_i(\mathbf{x}^r). \tag{18}$$




(a) Case one. (b) Case two.

FIG. 3. *The choice of subgradient* $\hat{\nabla}\phi_i(\mathbf{x}^r)$ *on nonsmooth point* $\mathbf{x}^r$.

**3.3. Choosing the optimal subgradients.** The subgradients in (16) and (17) should be chosen such that the steepest descent of $J$ in the set $\Gamma$ is always included in the convex feasible set $\mathcal{F}$.

Let $B(\mathbf{x}, r)$ denote the unit ball centered at $\mathbf{x} \in \mathbb{R}^n$ with radius $r$. At point $\mathbf{x}^r$, a search direction $v \in \partial B(\mathbf{0}, 1)$ is feasible if for all $i$, one of the three conditions hold:

- $\phi_i(\mathbf{x}^r) > 0$;
- $\phi_i(\mathbf{x}^r) = 0$ and there exists $d \in D\phi_i$ such that $v \cdot d \geq 0$;
- $\phi_i(\mathbf{x}^r) < 0$ and there exists $d \in D\phi_i$ such that $v \cdot d > 0$.

Define the set of feasible search directions as $C(\mathbf{x}^r)$, which is nonempty since we can choose $v$ to be $d/\|d\|$ for any nonzero $d \in D\phi_i$. $D\phi_i$ always contain a nonzero element by the first statement in Assumption 3. Then the direction of the steepest descent is $v^* := \arg\min_{v \in C(\mathbf{x}^r)} \nabla J \cdot v$. If $v^*$ is not unique, the tiebreaking mechanism is designed to be the following: choosing the one with the smallest first entry, the smallest second entry, and so on.[5] Then the optimal subgradient is chosen to be

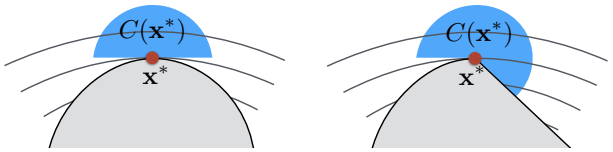
$$\hat{\nabla}\phi_i := \arg\min_{d \in DF_i} \nabla J \cdot d/\|d\|, \tag{19}$$

where $DF_i$ is the feasible set of subgradients for $\phi_i$ such that $DF_i := D\phi_i$ when $\phi_i > 0$; $DF_i := \{d \in D\phi_i | d \cdot v^* \geq 0\}$ when $\phi_i = 0$; and $DF_i := \{d \in D\phi_i | d \cdot v^* > 0\}$ when $\phi_i < 0$. The set $DF_i$ is nonempty by definition of $C(\mathbf{x}^r)$. To avoid singularity, let $d/\|d\|$ be $\mathbf{0}$ when $d = \mathbf{0}$. Figure 3 illustrates the above procedure in choosing the optimal subgradient, where the short arrow shows the direction of the steepest descent of $J$, the shaded sector shows the range of subdifferentials, the long arrow denotes the optimal subgradient, and the shaded half-space is the convex feasible set $\mathcal{F}_i$. In case one, $v^*$ is in the same direction of $\hat{\nabla}\phi_i$, while the two are perpendicular to each other in case two.

**4. Properties of the CFS algorithm.** This section shows the feasibility and convergence of Algorithm 1. The main result is summarized in the following theorem.

THEOREM 1 (convergence of Algorithm 1). *Under Algorithm* 1, *the sequence* $\{\mathbf{x}^{(k)}\}$ *will converge to some* $\mathbf{x}^* \in \Gamma$ *for any initial guess* $\mathbf{x}^{(0)}$ *such that* $\mathcal{F}^{(0)} \neq \emptyset$. $\mathbf{x}^*$ *is a strong local optimum of* (1) *if the limit is attained, i.e., there exists a constant* $K \in \mathbb{N}$ *such that* $\mathbf{x}^{(k)} = \mathbf{x}^*$ *for all* $k > K$. $\mathbf{x}^*$ *is at least a weak local optimum of* (1) *if the limit is not attained.*

[5] Note that the tiebreaking mechanism can be anything as long as it makes $v^*$ unique. The uniqueness is exploited in Proposition 3.




(a) A strong local optimum. (b) A weak local optimum.

FIG. 4. *Definition of local optima.*

We say that $\mathbf{x}^*$ is a strong local optimum of (1) if $J$ is nondecreasing along any feasible search direction, e.g., $\nabla J(\mathbf{x}^*)v \geq 0$ for all $v \in C(\mathbf{x}^*)$ as shown in Figure 4(a). We say that $\mathbf{x}^*$ is a weak local optimum of (1) if the KKT condition is satisfied for some subgradients, i.e., $\nabla J(\mathbf{x}^*) + \sum_{i=1}^{N} \lambda_i d_i = 0$ for some $d_i \in D\phi_i(\mathbf{x}^*)$ as shown in Figure 4(b). $\lambda_i$ is a Lagrange multiplier such that $\lambda_i \leq 0$ and $\lambda_i \phi_i(\mathbf{x}) = 0$ (complementary slackness) for all $i = 1, \ldots, N$. A strong local optimum is always a weak local optimum. The two are equivalent when all $\phi_i$'s are smooth at $\mathbf{x}^*$.

**4.1. Preliminary results.** Before proving Theorem 1, we present some preliminary results that are useful toward proving the theorem. Lemma 2 states that given a feasible reference point $\mathbf{x}^r$, $\mathcal{F}(\mathbf{x}^r)$ is a convex set containing $\mathbf{x}^r$ with nontrivial interior. This conclusion naturally leads to the hypothesis that a suboptimal reference $\mathbf{x}^r$ can be improved by optimizing $J$ in the convex feasible set $\mathcal{F}(\mathbf{x}^r)$. We will show in Proposition 3 that if the solution cannot be improved using Algorithm 1, then $\mathbf{x}^r$ is already a strong local optimum of (1). Otherwise, if we can keep improving the result using Algorithm 1, this process will generate a Cauchy sequence that converges to a weak local optimum of (1), which will be shown in Proposition 5. Given these results, the conclusion in the theorem follows naturally.

The interior of a set $S$ is denoted as $S^o$. We say that a reference point $\mathbf{x}^r \in \mathbb{R}^n$ is feasible if $\mathbf{x}^r \in \Gamma$, and $\mathbf{x}^* \in \Gamma$ is a fixed point of Algorithm 1 if

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{F}(\mathbf{x}^*)} J(\mathbf{x}). \tag{20}$$

LEMMA 2 (feasibility). *If $\mathbf{x}^r \in \Gamma$, then $\mathbf{x}^r \in \mathcal{F}(\mathbf{x}^r)$ and $\mathcal{F}^o(\mathbf{x}^r) \neq \emptyset$.*

*Proof.* When $\mathbf{x}^r$ is feasible, $\mathbf{x}^r \in \mathcal{F}_i(\mathbf{x}^r)$ for all $i$ according to the definitions in (15), (16), and (17). Hence $\mathbf{x}^r \in \mathcal{F}(\mathbf{x}^r)$.

*Claim* 1: *if $\mathbf{x}^r \in \Gamma^o$, then $\mathbf{x}^r \in \mathcal{F}^o(\mathbf{x}^r)$.* The condition $\mathbf{x}^r \in \Gamma^o$ implies that $\phi_i(\mathbf{x}^r) > 0$ for all $i = 1, \ldots, N$. Then the inequalities in (16) and (17) are not tight at $\mathbf{x}^r$. Hence $\mathbf{x}^r \in \mathcal{F}_i^o(\mathbf{x}^r)$ in either of the three cases. Since $N$ is finite, $\mathbf{x}^r \in \bigcap_{i=1}^{N} \mathcal{F}_i^o(\mathbf{x}^r) = \mathcal{F}^o(\mathbf{x}^r)$.

*Claim* 2: *if $\mathbf{x}^r \in \partial\Gamma$, there exists a nontrivial $v \in \mathbb{R}^n$ such that $\mathbf{x}^r + v \in \mathcal{F}^o(\mathbf{x}^r)$.* Let $I := \{i : \phi_i(x^r) = 0\}$. By the third statement in Assumption 3, there exists a unit vector $w \in \mathbb{R}^n$ such that $\partial_w \phi_i(\mathbf{x}^r) < 0$ for all $i \in I$. Fix $i \in I$. Let $a > 0$ be sufficiently small. When $\phi_i$ is concave,

$$\phi_i(\mathbf{x}^r - aw) \geq \partial_{-aw}\phi_i(\mathbf{x}^r) - \frac{a^2}{2} w^T H_i^* w \geq -a\partial_w \phi_i(\mathbf{x}^r) - \frac{a^2}{2} w^T H_i^* w > 0,$$

where the first inequality is due to semiconvexity, the second inequality is due to (7), and the third inequality is because $a$ is small. When $\phi_i$ is convex,

$$\phi_i(\mathbf{x}^r) + \hat{\nabla}\phi_i(\mathbf{x}^r) \cdot (-aw) \geq -a\partial_w \phi_i(\mathbf{x}^r) > 0,$$

where the first inequality is due to $\hat{\nabla}\phi_i(\mathbf{x}^r)\cdot w \le \partial_w\phi_i(\mathbf{x}^r)$ by definition (11). When $\phi_i$ is neither concave nor convex,

$$\phi_i(\mathbf{x}^r) + \hat{\nabla}\phi_i(\mathbf{x}^r)\cdot(-aw) \ge -a\partial_w\phi_i(\mathbf{x}^r) > \frac{a^2}{2}w^T H_i^* w.$$

Hence $\mathbf{x}^r - aw \in \mathcal{F}_i^o(\mathbf{x}^r)$ for any sufficiently small $a$. Since $I$ is finite, we can find a constant $\epsilon$ such that $\mathbf{x}^r - aw \in \mathcal{F}_i^o(\mathbf{x}^r)$ for all $i \in I$ and $0 < a \le \epsilon$. On the other hand, $\mathbf{x}^r \in \Gamma_j^o$ for all $j \notin I$. According to Claim 1, $\mathbf{x}^r \in \bigcap_{j\notin I}\mathcal{F}_j^o(\mathbf{x}^r)$. There exists a constant $\epsilon_0 > 0$ such that $B(\mathbf{x}^r,\epsilon_0) \subset \bigcap_{j\notin I}\mathcal{F}_j^o(\mathbf{x}^r)$. Define $v = -\min(\epsilon_0,\epsilon)w$. According to the previous discussion, $\mathbf{x}^r + v \in \mathcal{F}^o(\mathbf{x}^r)$.

Claims 1 and 2 imply that $\mathcal{F}(\mathbf{x}^r)$ has a nonempty interior. ∎

PROPOSITION 3 (fixed point). *If $\mathbf{x}^*$ is a fixed point of Algorithm 1, then $\mathbf{x}^*$ is a strong local optimum of (1).*

*Proof.* We need to show that $\nabla J(\mathbf{x}^*)\cdot v \ge 0$ for all $v \in C(\mathbf{x}^*)$. If $\nabla J(\mathbf{x}^*) = 0$, then $\mathbf{x}^*$ is the global optimum of (1). Consider the case $\nabla J(\mathbf{x}^*) \ne 0$. Claim that $\mathbf{x}^* \in \partial\Gamma$. First, since $\mathbf{x}^* \in \mathcal{F}(\mathbf{x}^*) \subset \Gamma$, $\mathbf{x}^*$ is a feasible point. Moreover, since $J$ is strictly convex, the optimal point $\mathbf{x}^*$ must be on the boundary of $\mathcal{F}(\mathbf{x}^*)$, i.e., $\mathbf{x}^* \in \partial\mathcal{F}_i(\mathbf{x}^*)$ for some $i$. According to (15), (16), and (17), $\partial\mathcal{F}_i(\mathbf{x}^*)$ equals to $\{\mathbf{x} : \phi_i(\mathbf{x}) = 0\}$ in Case 1, $\{\mathbf{x} : \phi_i(\mathbf{x}^*) + \hat{\nabla}\phi_i(\mathbf{x}^*)(\mathbf{x}-\mathbf{x}^*) = 0\}$ in Case 2, and $\{\mathbf{x} : \phi_i(\mathbf{x}^*) + \hat{\nabla}\phi_i(\mathbf{x}^*)(\mathbf{x}-\mathbf{x}^*) = \frac{1}{2}(\mathbf{x}-\mathbf{x}^*)^T H_i^*(\mathbf{x}-\mathbf{x}^*)\}$ in Case 3. Then $\mathbf{x}^* \in \partial\mathcal{F}_i(\mathbf{x}^*)$ implies that $\phi_i(\mathbf{x}^*) = 0$. Hence $\mathbf{x}^* \in \partial\Gamma$. Let $I = \{i : \phi_i(\mathbf{x}^*) = 0\} = \{i : \mathbf{x}^* \in \partial\mathcal{F}_i(\mathbf{x}^*)\}$.

Consider $v^* := \arg\min_{v\in C(\mathbf{x}^*)} \nabla J(\mathbf{x}^*)\cdot v$. If the minimum is not unique, use the tiebreaking mechanism discussed in subsection 3.3. Claim that $\hat{\nabla}\phi_i \cdot v^* \ge 0$ for all $i \in I$. For $i \in I$ such that $\phi_i$ is smooth at $\mathbf{x}^*$, the definition of $C(\mathbf{x}^*)$ implies that $\hat{\nabla}\phi_i(\mathbf{x}^*)\cdot v^* = \nabla\phi_i(\mathbf{x}^*)\cdot v^* \ge 0$. For $i \in I$ such that $\phi_i$ is not smooth at $\mathbf{x}^*$, $\hat{\nabla}\phi_i(\mathbf{x}^*)\cdot v^* \ge 0$ since $\hat{\nabla}\phi_i \in DF_i := \{d \in D\phi_i | d\cdot v^* \ge 0\}$. On the other hand, since $\mathbf{x}^*$ is the optimal solution of the smooth optimization $\min_{\mathbf{x}\in\mathcal{F}(\mathbf{x}^*)} J(\mathbf{x})$ , the KKT condition is satisfied,

$$\nabla J(\mathbf{x}^*) + \sum_{i=1}^{N}\lambda_i\hat{\nabla}\phi_i(\mathbf{x}^*) = 0.$$

The complementary slackness condition implies that $\lambda_i \le 0$ for $i \in I$ and $\lambda_j = 0$ for $j \notin I$. Hence

$$\nabla J(\mathbf{x}^*)\cdot v^* = -\sum_{i=1}^{N}\lambda_i\hat{\nabla}\phi_i(\mathbf{x}^*)\cdot v^* = -\sum_{i\in I}\lambda_i\hat{\nabla}\phi_i(\mathbf{x}^*)\cdot v^* \ge 0.$$

Thus $J(\mathbf{x}^*)\cdot v \ge 0$ for all $v \in C(\mathbf{x}^*)$. So $\mathbf{x}^*$ is a strong local optimum of (1). ∎

*Remark* 1. Lemma 2 and Proposition 3 imply that a feasible $\mathbf{x}^r$ can always be improved by optimizing over the convex feasible set $\mathcal{F}(\mathbf{x}^r)$ if $\mathbf{x}^r$ itself is not a strong local optimum. However, the existence of a nonempty convex feasible set for an infeasible reference point is more intricate, which is deeply related to the choice of the functions $\phi_i$'s. The design considerations of $\phi_i$'s such that a nonempty convex feasible set always exists will be addressed in section 5.

LEMMA 4 (strong descent). *For any feasible $\mathbf{x}^{(k)}$, the descent of the objective function satisfies that $\nabla J(\mathbf{x}^{(k+1)})(\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}) \ge 0$. Moreover, if $J(\mathbf{x}^{(k+1)}) = J(\mathbf{x}^{(k)})$, then $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}$.*

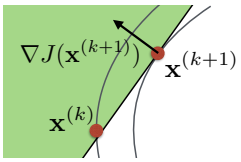



FIG. 5. *The descent of the sequence* $\{\mathbf{x}^{(k)}\}$.

*Proof.* Claim that $\mathcal{F}(\mathbf{x}^{(k)})$ is a subset of the half space $H := \{\mathbf{x} \mid \nabla J(\mathbf{x}^{(k+1)}) \cdot (\mathbf{x} - \mathbf{x}^{(k+1)}) \geq 0\}$ as shown by the shaded area in Figure 5. If not, there must be some $\hat{\mathbf{x}} \in \mathcal{F}(\mathbf{x}^{(k)})$ such that $\nabla J(\mathbf{x}^{(k+1)}) \cdot (\hat{\mathbf{x}} - \mathbf{x}^{(k+1)}) < 0$. Let $v := \hat{\mathbf{x}} - \mathbf{x}^{(k+1)}$. Since $\mathcal{F}(\mathbf{x}^{(k)})$ is convex, then $\mathbf{x}^{(k+1)} + av \in \mathcal{F}(\mathbf{x}^{(k)})$ for $a \in [0, 1]$. Since $J$ is smooth, there exists a positive constant $c > 0$ such that $J(\mathbf{x}^{(k+1)} + av) \leq J(\mathbf{x}^{(k+1)}) + a\nabla J(\mathbf{x}^{(k+1)}) \cdot v + ca^2\|v\|^2$. When $a$ is sufficiently small, the right-hand side of the inequality is strictly smaller than $J(\mathbf{x}^{(k+1)})$. Then $J(\mathbf{x}^{(k+1)} + av) < J(\mathbf{x}^{(k+1)})$, which contradicts the fact that $\mathbf{x}^{(k+1)}$ is the minimum of $J$ in the convex feasible set $\mathcal{F}(\mathbf{x}^{(k)})$. Hence the claim is true. Since $\mathbf{x}^{(k)} \in \mathcal{F}(\mathbf{x}^{(k)}) \subset H$, then $\nabla J(\mathbf{x}^{(k+1)})(\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}) \geq 0$. Moreover, since $J$ is strictly convex, $J(\mathbf{x}) > J(\mathbf{x}^{(k+1)})$ for all $\mathbf{x} \in H \setminus \{\mathbf{x}^{(k+1)}\}$. Hence $J(\mathbf{x}^{(k+1)}) = J(\mathbf{x}^{(k)})$ implies $\mathbf{x}^{(k)} = \mathbf{x}^{(k+1)}$. ∎

PROPOSITION 5 (convergence of strictly descending sequence). *Consider the sequence* $\{\mathbf{x}^{(k)}\}$ *generated by Algorithm* 1. *If* $J(\mathbf{x}^{(1)}) > J(\mathbf{x}^{(2)}) > \cdots$, *then the sequence* $\{\mathbf{x}^{(k)}\}$ *converges to a weak local optimum* $\mathbf{x}^*$ *of* (1).

*Proof.* The monotone sequence $\{J(\mathbf{x}^{(k)})\}_{i=2}^{\infty}$ converges to some value $a \geq \min J$. If $a = \min J$, the sequence $\{\mathbf{x}^{(k)}\}$ converges to the global optima. Consider the case $a > \min J$. Since $J$ is strictly convex by Assumption 1, the set $\{\mathbf{x} : J(\mathbf{x}) \leq J(\mathbf{x}^{(1)})\}$ is compact. Then there exists a subsequence of $\{\mathbf{x}^{(k)}\}$ that converges to $\mathbf{x}^*$ such that $J(\mathbf{x}^*) = a$. Since $\Gamma$ is closed, $\mathbf{x}^* \in \Gamma$.

We need to show that the whole sequence $\{\mathbf{x}^{(k)}\}$ converges to $\mathbf{x}^*$. Suppose not, then there exists $\delta > 0$ such that for all $K > 0$, there exists $k > K$ s.t. $\|\mathbf{x}^{(k)} - \mathbf{x}^*\| \geq \delta$. For any $\epsilon \in (0, \delta)$, there exists $k, j \in \mathbb{N}$ such that $\|\mathbf{x}^{(k)} - \mathbf{x}^*\| \leq \epsilon$ and $\|\mathbf{x}^{(k+j)} - \mathbf{x}^*\| \geq \delta$. Since $J$ is strictly convex, there exists $c > 0$ such that

$$\begin{aligned}
J(\mathbf{x}^{(k)}) &\geq J(\mathbf{x}^{(k+1)}) + \nabla J(\mathbf{x}^{(k+1)})(\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}) + c\|\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\|^2 \\
&\geq J(\mathbf{x}^{(k+1)}) + c\|\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\|^2 \\
&\geq J(\mathbf{x}^{(k+j)}) + c\|\mathbf{x}^{(k+j-1)} - \mathbf{x}^{(k+j)}\|^2 + \cdots + \|\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\|^2 \\
&\geq J(\mathbf{x}^{(k+j)}) + c\|\mathbf{x}^{(k)} - \mathbf{x}^{(k+j)}\|^2 \\
&\geq J(\mathbf{x}^{(k+j)}) + c(\|\mathbf{x}^{(k)} - \mathbf{x}^*\| - \|\mathbf{x}^{(k+j)} - \mathbf{x}^*\|)^2 \\
&\geq a + c(\delta - \epsilon)^2,
\end{aligned}$$

which contradicts the fact that $J(\mathbf{x}^{(k)}) \to a$ as $\epsilon \to 0$. Note that the second inequality is due to $\nabla J(\mathbf{x}^{(k+1)})(\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}) \geq 0$ in Lemma 4. The third inequality is by induction. The fourth inequality and the fifth inequality follow from $\Delta$-inequality. Hence we conclude that $\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}^*$.

Then we need to show that $\mathbf{x}^*$ is a weak local optimum. The proof can be divided into two steps. First, we show that there is a subsequence of the convex feasible sets $\{\mathcal{F}^{(k)}\}$ that converges pointwise to a suboptimal convex feasible set $\mathcal{G}$ at point $\mathbf{x}^*$. Suboptimality of $\mathcal{G}$ means that the subgradients are not chosen according to subsection 3.3. Then we show that $\mathbf{x}^*$ is the minimum of $J$ in $\mathcal{G}$. Thus the KKT condition is satisfied at $\mathbf{x}^*$ and $\mathbf{x}^*$ is a weak local optimum of (1).

Consider any $\phi_i$. If $\phi_i$ is smooth at $\mathbf{x}^*$, then it is smooth in a neighborhood of $\mathbf{x}^*$ as $\phi_i$ is assumed to be piecewise smooth. Then $\hat{\nabla}\phi_i(\mathbf{x}^{(k)})$ converges to $d_i := \nabla\phi_i(\mathbf{x}^*)$. If $\phi_i$ is not smooth at $\mathbf{x}^*$, it is still locally Lipschitz due to semiconvexity. Then $\hat{\nabla}\phi_i$ is locally bounded. Hence there is a subsequence $\{\hat{\nabla}\phi_i(\mathbf{x}^{(k_j)})\}_{k_j\in\mathbb{N}}$ that converges to some $d_i \in \mathbb{R}^n$. Claim that $d_i \in D\phi_i(\mathbf{x}^*)$. By definition (11), for any $v \in \mathbb{R}^n$, $\hat{\nabla}\phi_i(\mathbf{x}^{(k_j)}) \cdot v \le \partial_v\phi_i(\mathbf{x}^{(k_j)})$. Since $\phi_i$ is piecewise smooth, then either $\partial_v\phi_i(\mathbf{x}^{(k_j)}) \to \partial_v\phi_i(\mathbf{x}^*)$ or $\partial_v\phi_i(\mathbf{x}^{(k_j)}) \to -\partial_{-v}\phi_i(\mathbf{x}^*)$. Since $-\partial_{-v}\phi_i(\mathbf{x}^*) \le \partial_v\phi_i(\mathbf{x}^*)$ by (7), we have the following inequality:

$$d_i \cdot v = \lim_{k_j\to\infty} \hat{\nabla}\phi_i(\mathbf{x}^{(k_j)}) \cdot v \le \lim_{k_j\to\infty} \partial_v\phi_i(\mathbf{x}^{(k_j)}) \le \partial_v\phi_i(\mathbf{x}^*).$$

Hence by definition (11), $d_i \in D\phi_i(\mathbf{x}^*)$. Then we can choose a subsequence $\{\mathbf{x}^{(k_n)}\}_{k_n\in\mathbb{N}}$ such that $\phi_i(\mathbf{x}^{(k_n)})$ converges to $\phi_i(\mathbf{x}^*)$ and $\hat{\nabla}\phi_i(\mathbf{x}^{(k_n)})$ converges to $d_i \in D\phi_i(\mathbf{x}^*)$ for all $i = 1, \dots, N$. For simplicity and without loss of generality, we use the same notation for the subsequence as the original sequence in the following discussion.

Define a new convex feasible set $\mathcal{G}_i$ such that $\mathcal{G}_i = \Gamma_i$ if $\phi_i$ is concave, $\mathcal{G}_i = \{\mathbf{x} : \phi_i(\mathbf{x}^*) + d_i(\mathbf{x} - \mathbf{x}^*) \ge 0\}$ if $\phi_i$ is convex, and $\mathcal{G}_i = \{\mathbf{x} : \phi_i(\mathbf{x}^*) + d_i(\mathbf{x} - \mathbf{x}^*) \ge \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T H_i^*(\mathbf{x} - \mathbf{x}^*)\}$ otherwise. Let $\mathcal{G} = \bigcap_{i=1}^N \mathcal{G}_i$.

Claim that $\mathcal{G} = \lim_{k\to\infty} \mathcal{F}^{(k)}$, where $\lim_{k\to\infty} \mathcal{F}^{(k)} := \bigcap_{k\to\infty} \bigcup_{j=k}^{\infty} \mathcal{F}^{(k)}$. Consider $z \in \mathcal{G}$. If $\phi_i$ is concave, then $z \in \mathcal{F}_i^{(k)} := \mathcal{F}_i(\mathbf{x}^{(k)})$ for all $k$ according to (16). If $\phi_i$ is convex, then

$$\lim_{k\to\infty} \left[\phi_i(\mathbf{x}^{(k)}) + \hat{\nabla}\phi_i(\mathbf{x}^{(k)})(z - \mathbf{x}^{(k)})\right] = \phi_i(\mathbf{x}^*) + d_i(z - \mathbf{x}^*) \ge 0,$$

which implies that $z \in \lim_{k\to\infty} \mathcal{F}_i^{(k)}$. Similarly, we can show that if $\phi_i$ is neither convex nor concave, $z$ also lies in the limit of $\mathcal{F}_i^{(k)}$. Hence $z \in \bigcap_{i=1}^N \lim_{k\to\infty} \mathcal{F}_i^{(k)} = \lim_{k\to\infty} \mathcal{F}^{(k)}$. Since $z$ is arbitrary, $\mathcal{G} \subset \lim_{k\to\infty} \mathcal{F}^{(k)}$. For any $z \in \lim_{k\to\infty} \mathcal{F}^{(k)}$, then there exists a sequence $\{z_k\}$ that converges to $z$ such that $z_k \in \mathcal{F}^{(k)}$. For any $i$, if $\phi_i$ is concave, then $z_k \in \mathcal{F}_i^{(k)} = \mathcal{G}_i$. Since $\mathcal{G}_i$ is closed, $z \in \mathcal{G}_i$. If $\phi_i$ is convex, then

$$\phi_i(\mathbf{x}^*) + d_i(z - \mathbf{x}^*) = \lim_{k\to\infty} \left[\phi_i(\mathbf{x}^{(k)}) + \hat{\nabla}\phi_i(\mathbf{x}^{(k)})(z_k - \mathbf{x}^{(k)})\right] \ge 0,$$

which implies that $z \in \mathcal{G}_i$. Similarly, we can show that $z \in \mathcal{G}_i$ if $\phi_i$ is neither convex nor concave. Hence $z \in \mathcal{G}$. And we verify that $\mathcal{G} = \lim_{k\to\infty} \mathcal{F}^{(k)}$.

Claim that $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{G}} J(\mathbf{x})$. Suppose not, then there exists $z \in \mathcal{G}$ such that $J(z) < J(\mathbf{x}^*)$. For all $\epsilon > 0$, there exists $y \in \mathcal{F}^{(k)}$ for some $k \in \mathbb{N}$ such that $\|z - y\| < \epsilon$. Since $J$ is smooth, then $J(y) - J(z) < O(\epsilon)$. Thus $J(y) < J(\mathbf{x}^*)$ when $\epsilon$ is sufficiently small. This contradicts $J(y) > J(\mathbf{x}^{(k)}) > J(\mathbf{x}^*)$. Hence $J(z) \ge J(\mathbf{x}^*)$ for all $z \in \mathcal{G}$. If there exists $z \in \mathcal{G} \setminus \{\mathbf{x}^*\}$ such that $J(z) = J(\mathbf{x}^*)$, then $\frac{z+\mathbf{x}^*}{2} \in \mathcal{G}$ and $J(\frac{z+\mathbf{x}^*}{2}) < \frac{J(z)+J(\mathbf{x}^*)}{2} = J(\mathbf{x}^*)$ since $\mathcal{G}$ is convex and $J$ is strictly convex. However, this contradicts the conclusion that $J(z) \ge J(\mathbf{x}^*)$ for all $z \in \mathcal{G}$. Hence $\mathbf{x}^*$ is the unique minimum of $J$ in the set $\mathcal{G}$. And the KKT condition is satisfied, i.e., $\nabla J(\mathbf{x}^*) + \sum_{i=1}^N \lambda_i d_i = 0$ for $d_i \in D\phi_i(\mathbf{x}^*)$. Then $\mathbf{x}^*$ is a weak local optimum.

It is worth noting that if all $\phi_i$'s are smooth at $\mathbf{x}^*$, $\mathcal{G} = \mathcal{F}(\mathbf{x}^*)$. Then $\mathbf{x}^*$ is a fixed point, and thus a strong local optimum by Proposition 3. ∎

*Remark* 2. Lemma 4 and Proposition 5 justify the adoption of the terminate condition (14), which is indeed equivalent to the standard terminate condition (13), e.g., convergence in the objective function implies convergence in the solution.

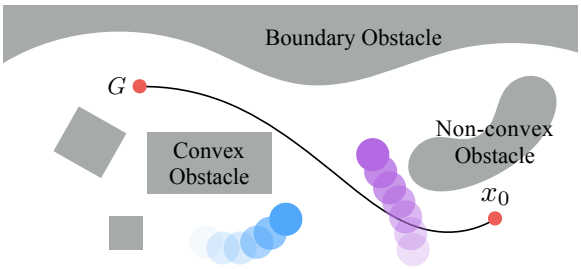



FIG. 6. *The motion planning problem.*

### 4.2. Proof of the main result.

*Proof of Theorem* 1. If $\mathcal{F}^{(0)}$ is nonempty, then $\mathbf{x}^{(1)} \in \Gamma$ can be obtained by solving the convex optimization (12). By Lemma 2, $\mathcal{F}^{(1)}$ has a nonempty interior, then $\mathbf{x}^{(2)} \in \Gamma$ can be obtained. By induction, we can conclude that $\mathbf{x}^{(i)} \in \mathcal{F}^{(i-1)} \subset \Gamma$ for $i = 1, 2, 3, \ldots$. Moreover, as a better solution is found at each iteration, then $J(\mathbf{x}^{(1)}) \geq J(\mathbf{x}^{(2)}) \geq \ldots$. This leads to two cases. The first case is that $J(\mathbf{x}^{(K)}) = J(\mathbf{x}^{(K+1)})$ for some $K$, while the second case is that the cost keeps decreasing strictly, i.e., $J(\mathbf{x}^{(1)}) > J(\mathbf{x}^{(2)}) > \ldots$. In the first case, the condition $J(\mathbf{x}^{(K)}) = J(\mathbf{x}^{(K+1)})$ is equivalent to $\mathbf{x}^{(K)} = \mathbf{x}^{(K+1)}$ by Lemma 4. By induction, the algorithm converges in the sense that $\mathbf{x}^{(k)} = \mathbf{x}^{(k+1)}$ and $J(\mathbf{x}^{(k)}) = J(\mathbf{x}^{(k+1)})$ for all $k \geq K$. Moreover, as $\mathbf{x}^* := \mathbf{x}^{(K)}$ is a fixed point, it is a strong local optimum by Proposition 3. If the cost keeps decreasing, e.g., $J(\mathbf{x}^{(1)}) > J(\mathbf{x}^{(2)}) > \ldots$, then the sequence $\{\mathbf{x}^{(k)}\}$ converges to a weak local optimum $\mathbf{x}^*$ by Proposition 5. ∎

**5. Application on motion planning for mobile robots.** In this section, Algorithm 1 is applied to a motion planning problem for mobile robots [24]. Its application to other systems can be found in [14]. The problem will be formulated in subsection 5.1 and then transformed into the benchmark form (1) and (2). The major difficulty in transforming the problem lies in finding the semiconvex function $\phi_i$ to describe the constraints. The method to construct $\phi_i$ in two dimensions is discussed in subsection 5.2 and the resulting convex feasible set is illustrated through examples in subsection 5.3. The performance of Algorithm 1 is shown in subsection 5.4 and compared ITP and SQP.

**5.1. Problem statement.** Suppose a mobile robot needs to plan a trajectory in $\mathbb{R}^2$ from the start point $x_0$ to the goal point $G$ as shown in Figure 6. Let $x_q \in \mathbb{R}^2$ be the position of the robot at time step $q$. Define the decision variable as $\mathbf{x} := \begin{bmatrix} x_1^T & \cdots & x_h^T \end{bmatrix}^T \in \mathbb{R}^{2h}$, where $h$ is the planning horizon. The whole trajectory is denoted as $\bar{\mathbf{x}} := \begin{bmatrix} x_0^T & \mathbf{x}^T & G^T \end{bmatrix}^T \in \mathbb{R}^{2(h+2)}$. Define a sequence of selection functions $l_q : \mathbb{R}^{2h} \to \mathbb{R}^2$ as $x_q = l_q(\mathbf{x})$. The sampling time is $t_s$. Let the velocity at $q$ be $v_q := \frac{x_q - x_{q-1}}{t_s}$ and the acceleration at $q$ be $a_q := \frac{v_q - v_{q-1}}{t_s}$.

*The cost function.* The cost function of the problem is designed as

$$J(\mathbf{x}) = w_1 \|\mathbf{x} - \mathbf{x}^r\|_Q^2 + w_2 \|\mathbf{x}\|_S^2, \tag{21}$$

where $w_1, w_2 \in \mathbb{R}^+$. The first term $\|\mathbf{x} - \mathbf{x}^r\|_Q^2 := (\bar{\mathbf{x}} - \bar{\mathbf{x}}^r)^T Q (\bar{\mathbf{x}} - \bar{\mathbf{x}}^r)$ penalizes the distance from the target trajectory to the reference trajectory. The second term $\|\mathbf{x}\|_S^2 := \bar{\mathbf{x}}^T S \bar{\mathbf{x}}$ penalizes the properties of the target trajectory itself, e.g., length of the trajectory and magnitude of acceleration. The matrices $Q, S \in \mathbb{R}^{2(h+2)\times 2(h+2)}$ can

be constructed from the following components: (1) matrix for position $Q_1 := I_{2(h+2)}$; (2) matrix for velocity $Q_2 := V^T V$, and (3) matrix for acceleration $Q_3 := A^T A$. Note that $V \in \mathbb{R}^{2(h+1)\times 2(h+2)}$ and $A \in \mathbb{R}^{2h\times 2(h+2)}$ are defined as

$$V = \frac{1}{t_s}\begin{bmatrix} I_2 & -I_2 & 0 & \cdots & 0 \\ 0 & I_2 & -I_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & I_2 & -I_2 \end{bmatrix}, A = \frac{1}{t_s^2}\begin{bmatrix} I_2 & -2I_2 & I_2 & 0 & \cdots & 0 \\ 0 & I_2 & -2I_2 & I_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & I_2 & -2I_2 & I_2 \end{bmatrix},$$

which take finite differences of the trajectory $\mathbf{x}$ such that $V\mathbf{x}$ returns the velocity vector and $A\mathbf{x}$ returns the acceleration vector. Then $Q := \sum_{i=1}^{3} c_i^q Q_i$ and $S := \sum_{i=1}^{3} c_i^s Q_i$, where $c_i^q$ and $c_i^s$ are positive constants. Assumption 1 is satisfied.

*The constraints.* The obstacles in the environment are denoted as $\mathcal{O}_j \in \mathbb{R}^2$ for $j \in \mathbb{N}$. Each $\mathcal{O}_j$ is simply connected and open with a piecewise smooth boundary that does not contain any sharp concave corner.[6] For example, there are seven such obstacles in Figure 6, where five of them are static and two are dynamic. Let $\mathcal{O}_j$ denote obstacle $j$ when centered at the origin. The area occupied by obstacle $j$ at time step $q$ is defined as $\mathcal{O}_j(q)$. Define a linear isometry $\mathcal{T}_{j,q} : \mathbb{R}^2 \to \mathbb{R}^2$ such that $\mathcal{O}_j = \mathcal{T}_{j,q}(\mathcal{O}_j(q))$. Then $x_q \notin \mathcal{O}_j(q)$ is equivalent to $\mathcal{T}_{j,q}(x_q) \notin \mathcal{O}_j$. Hence the constraint for the optimization is

$$\mathbf{x} \in \Gamma := \{\mathbf{x} \in \mathbb{R}^{2h} : \mathcal{T}_{j,q}(l_q(\mathbf{x})) \notin \mathcal{O}_j \ \forall j, \forall q = 1, \ldots, h\}. \tag{22}$$

It is easy to verify that Assumption 2 is satisfied.

**5.2. Transforming the problem.** In order to apply Algorithm 1, a semiconvex decomposition (2) satisfying Assumption 3 needs to be performed. For example, obstacles containing concave corners need to be partitioned into several overlapping obstacles that do not have concave corners as discussed in subsection 2.2. Without loss of generality, $\mathcal{O}_j$ is assumed to represent obstacles after decomposition such that it is either a convex obstacle, a boundary obstacle, or a nonconvex obstacle as shown in Figure 6. For each $\mathcal{O}_j$, we first construct a simple function $\varphi_j : \mathbb{R}^2 \to \mathbb{R}$ and then use $\varphi_j$ to construct $\phi_i$. The function $\varphi_j$ is continuous, piecewise smooth, and semiconvex such that $\mathcal{O}_j = \{x \in \mathbb{R}^2 : \varphi_j(x) < 0\}$ and $\partial\mathcal{O}_j = \{x \in \mathbb{R}^2 : \varphi_j(x) = 0\}$. We call $\varphi_j$ a safety index in the following discussion, since it typically measures the distance to an obstacle. The construction of $\varphi_j$ in each case is discussed below.

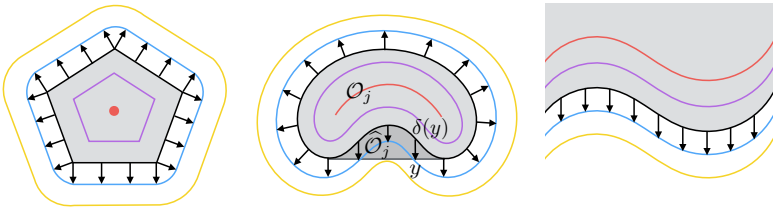
*Convex obstacle.* A convex obstacle refers to the case that $\mathcal{O}_j$ is compact and convex. In this case, $\varphi_j$ is defined to be the signed distance function to $\mathcal{O}_j$, i.e.,

$$\varphi_j(x) := \begin{cases} \min_{y\in\partial\mathcal{O}_j} \|x - y\|, & x \notin \mathcal{O}_j, \\ -\min_{y\in\partial\mathcal{O}_j} \|x - y\|, & x \in \mathcal{O}_j. \end{cases} \tag{23}$$

*Boundary obstacle.* A boundary obstacle refers to a noncompact $\mathcal{O}_j$ such that there is an affine parameterization of $\partial\mathcal{O}_j$, e.g., if we rotate and align the obstacle properly, there exists a continuous and piecewise smooth semiconvex function $f : \mathbb{R} \to \mathbb{R}$ such that $p_2 = f(p_1)$ describes $\partial\mathcal{O}_j$, where $x = (p_1, p_2) \in \mathbb{R}^2$. Then $\varphi_j$ is defined as the directional distance along $(0, 1)$ to the boundary $\partial\mathcal{O}_j$, i.e.,

$$\varphi_j(x) := f(p_1) - p_2. \tag{24}$$

[6] Note that the obstacles may not be physical, but denote the infeasible area in the state space. A sharp concave corner in $\mathbb{R}^2$ is a concave corner with a 360° tangent angle. The existence of a sharp concave corner will violate Assumption 2, since there does not exist a two-dimensional convex neighborhood in $\mathcal{O}_j^c$ at a sharp concave corner.




(a) Convex obstacle. (b) Concave obstacle. (c) Boundary obstacle.

FIG. 7. *Contours of the safety indices for typical obstacles.*

*Nonconvex obstacle.* A nonconvex obstacle refers to the case that $\mathcal{O}_j$ is compact, but nonconvex. As the signed distance function for a nonconvex set is not semiconvex, the strategy is to introduce directional distance. Denote the smallest convex envelop of the obstacle $\mathcal{O}_j$ as $\widehat{\mathcal{O}}_j$. Then $\varphi_j$ is defined as a signed directional distance function which computes the minimum distance from a point $x$ to $\partial\mathcal{O}_j$ in the direction that is perpendicular to $\partial\widehat{\mathcal{O}}_j$ as shown in Figure 7(b). Let $\delta$ be a correspondence function that maps a point $y \in \partial\widehat{\mathcal{O}}_j$ to the closest $z \in \partial\mathcal{O}_j$ such that $z - y$ is perpendicular to $\partial\widehat{\mathcal{O}}_j$ at $y$. It is assumed that $\delta$ is bijective and $\|y - \delta(y)\|$ is semiconvex in $y$. Then

$$\varphi_j(x) = \begin{cases} \min_{y\in\partial\widehat{\mathcal{O}}_j}[\|x-y\| + \|y-\delta(y)\|] & x \notin \widehat{\mathcal{O}}_j, \\ -\min_{y\in\partial\widehat{\mathcal{O}}_j}[\|x-y\| - \|y-\delta(y)\|] & x \in \widehat{\mathcal{O}}_j. \end{cases} \tag{25}$$

It is easy to verify that $\varphi_j$ in all three cases are continuous, piecewise smooth, and semiconvex and satisfy the first two arguments in Assumption 3. Moreover, $\varphi_j$ is strictly convex for convex obstacles. According to (23), (24), and (25), the contours of the safety indices are shown in Figure 7. Based on the safety indices, (22) can be rewritten as

$$\mathbf{x} \in \Gamma = \bigcap_{j,q} \{\mathbf{x} \in \mathbb{R}^{2h} : \varphi_j(\mathcal{T}_{j,q}(l_q(\mathbf{x}))) \geq 0\}. \tag{26}$$

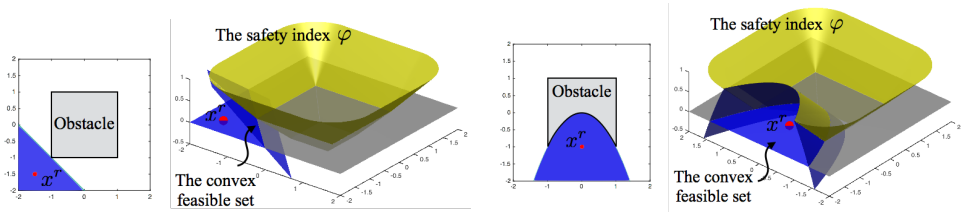
Define $\phi_{j,q}(\mathbf{x}) := \varphi_j(\mathcal{T}_{j,q}(l_q(\mathbf{x})))$ for all $j$ and $q$. Let $\Gamma_{j,q} := \{\mathbf{x} \in \mathbb{R}^{2h} : \phi_{j,q}(\mathbf{x}) \geq 0\}$. When the obstacles do not overlap with each other at every time step as shown in Figure 6, the third argument in Assumption 3 is satisfied. Then $\{\phi_{j,q}\}_{j,q}$ is a decomposition of $\Gamma$ that satisfies Assumption 3. The convex feasible set $\mathcal{F} := \bigcap_{j,q} \mathcal{F}_{j,q}$ can be constructed according to the discussion in subsection 3.2. And Algorithm 1 can be applied.

**5.3. The convex feasible set—Examples.** This section illustrates the configuration of convex feasible sets with examples. Since $\phi_{j,q}$ only depends on $x_q$, $\mathcal{F}_{j,q}(\mathbf{x}^r)$ only constraints $x_q$. For example, according to (16), the convex feasible set for a convex $\phi_{j,q}$ is

$$\mathcal{F}_{j,q}(\mathbf{x}^r) = \{\mathbf{x} : \varphi_j(\mathcal{T}_{j,q}(x_q^r)) + \hat{\nabla}\varphi_j(\mathcal{T}_{j,q}(x_q^r))\nabla\mathcal{T}_{j,q}(x_q^r)(x_q - x_q^r) \geq 0\}. \tag{27}$$

For simplicity, define $\mathcal{F}_{j,q}^q(\mathbf{x}^r) := l_q(\mathcal{F}_{j,q}(\mathbf{x}^r)) \in \mathbb{R}^2$. In the following discussion, we will first illustrate $\mathcal{F}_{j,q}^q(\mathbf{x}^r)$ in $\mathbb{R}^2$ and then $\bigcap_q \mathcal{F}_{j,q}(\mathbf{x}^r)$ in $\mathbb{R}^{2h}$.

*The convex feasible set in* $\mathbb{R}^2$. We illustrate the convex feasible set for one obstacle at one time step. For simplicity, subscripts $j$ and $q$ are removed and $\mathcal{T}_{j,q}$ is assumed




(a) Convex obstacle. (b) Nonconvex obstacle.

FIG. 8. *Illustration of the convex feasible set in* $\mathbb{R}^2$.

to be identity. Consider a polygon obstacle with vertices $a = (1,1)$, $b = (-1,1)$, $c = (-1,-1)$, and $d = (1,-1)$ as shown in Figure 8(a). Let $\|x\|_\infty$ denote the $l_\infty$ norm, i.e., $\|x\|_\infty := \max\{|p_1|, |p_2|\}$. Then according to (23),

$$\varphi(x) = \begin{cases} \max\{-1-p_1, p_1-1, p_2-1, -1-p_2\}, & \|x\|_\infty \le 1, \\ \min\{\|x-a\|, \|x-b\|, \|x-c\|, \|x-d\|\}, & |p_1|>1, |p_2|>1, \\ \min\{|p_1-1|, |p_1+1|\}, & |p_1|>1, |p_2|<1, \\ \min\{|p_2-1|, |p_2+1|\}, & |p_1|<1, |p_2|>1. \end{cases} \tag{28}$$

For a reference point $x^r = (-1.5, -1.5)$, $\varphi(x^r) = \frac{\sqrt{2}}{2}$, and $\nabla\varphi(x^r) = \frac{\sqrt{2}}{2}[-1,-1]$. The convex feasible set[7] is $\mathcal{F}(x^r) = \{x : \varphi(x^r) + \nabla\varphi(x^r)(x-x^r) \ge 0\} = \{x : \frac{\sqrt{2}}{2}(-p_1 - p_2 + 2) \ge 0\}$. The bowl-shaped surface in Figure 8(a) illustrates the safety index $\varphi$. The plane that is tangent to the safety index satisfies the function $\frac{\sqrt{2}}{2}(-p_1 - p_2 + 2) = 0$. Since $\varphi$ is convex, the tangent plane is always below $\varphi$. The convex feasible set can be regarded as the projection of the positive portion of the tangent plane onto the zero level set.

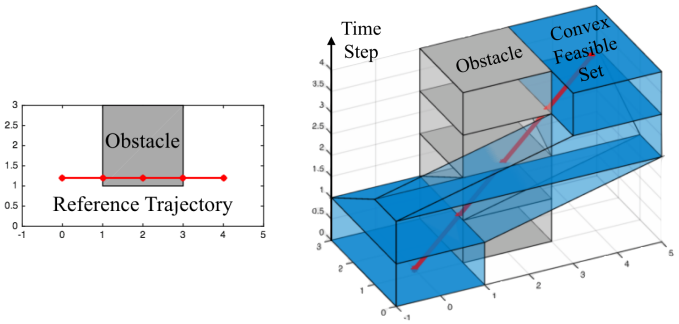
Consider the case that points $c$ and $d$ are not connected by a straight line, but a concave curve $p_2 = -(p_1)^2$ as shown in Figure 8(b). Then according to (25),

$$\varphi(x) = \begin{cases} \max\{-(p_1)^2 - p_2, p_1-1, p_2-1, -1-p_2\}, & \|x\|_\infty \le 1, p_2 \ge -(p_1)^2, \\ \min\{\|x-a\|, \|x-b\|, \|x-c\|, \|x-d\|\}, & |p_1|>1, |p_2|>1, \\ \min\{|p_1-1|, |p_1+1|\}, & |p_1|>1, |p_2|<1, \\ p_2 - 1, & p_2 > 1, \\ -(p_1)^2 - p_2, & p_2 < -(p_1)^2. \end{cases} \tag{29}$$

The hessian of $\varphi$ is bounded below by $-H^* = [-2, 0; 0, 0]$. For a reference point $x^r = (0,-1)$, $\varphi(x^r) = 1$ and $\nabla\varphi(x^r) = [0,-1]$. The convex feasible set is $\mathcal{F}(x^r) = \{x : \varphi(x^r) + \nabla\varphi(x^r)(x-x^r) \ge \frac{1}{2}(x-x^r)H^*(x-x^r)\} = \{x : -p_2 - (p_1)^2 \ge 0\}$. The bowl-shaped surface in Figure 8(b) illustrates the safety index $\varphi$. The parabolic surface that is tangent to the safety index represents the function $-p_2 - (p_1)^2 = 0$. The convex feasible set is the projection of the positive portion of the surface onto the zero level set.

*The convex feasible set in higher dimension.* We illustrate the convex feasible set for one obstacle over the entire time horizon. The obstacle is shown in Figure 9(a), which is a translated version of the obstacle in Figure 8(a). The reference trajectory

[7] When implementing Algorithm 1 in software, there is no need to explicitly compute the safety indices as shown in the examples. The solver just needs to know the rules (23), (24), and (25) in computing those indices. The gradients or hessians of the safety indices can be computed numerically.




(a) The reference trajectory. (b) The convex feasible set over time.

FIG. 9. *Illustration of the convex feasible set in* $\mathbb{R}^{2h}$.

violates the obstacle avoidance constraint. Figure 9(b) shows the convex feasible sets computed for each time step. Those sets formulate a corridor around the time-augmented obstacle. A new trajectory will be computed in the corridor. Although the projection of the corridor into $\mathbb{R}^2$ is not convex, each time slice of the corridor is convex. In $\mathbb{R}^{2h}$, those slices are stuck together orthogonally, hence formulating a convex subset of $\mathbb{R}^{2h}$.

As pointed out in Remark 1, for an infeasible reference trajectory, when there are multiple obstacles, the existence of a corridor that bypasses all time-augmented obstacles under the proposed algorithm is hard to guarantee. In Lemma 6, we show that a convex feasible set has a nonempty interior for any reference trajectory if certain geometric conditions are satisfied.

LEMMA 6 (feasibility). *If all obstacles are convex and have disjoint closures, i.e.,* $\bar{\mathcal{O}}_i(q) \bigcap \bar{\mathcal{O}}_j(q) = \emptyset$ *for all* $q$ *and* $i \neq j$*, then the convex feasible set* $\mathcal{F}(\mathbf{x}^r)$ *has a nonempty interior for all* $\mathbf{x}^r \in \mathbb{R}^n$ *when* $\phi_{j,q}$ *is chosen according to* (23) *and* (26).

*Proof.* Considering Lemma 2, we only need to prove that $\mathcal{F}(\mathbf{x}^r)$ has a nonempty interior for infeasible $\mathbf{x}^r$. The infeasibility of $\mathbf{x}^r$ implies that some $\phi_{j,q}(\mathbf{x}^r)$ is negative. Fixing $q$, since $\bar{\mathcal{O}}_j(q)$'s are disjoint, only one $\phi_{j,q}$ can be negative. Without loss of generality, suppose $\phi_{1,q}(\mathbf{x}^r) < 0$ and $\phi_{j,q}(\mathbf{x}^r) \geq 0$ for $j \geq 2$. Since the safety index $\varphi_j$ is a signed distance function in (23) and $\mathcal{T}_{j,q}$ is an isometry, $\|\hat{\nabla}\varphi_j\| = 1$ and $\|\mathcal{T}_{j,q}\| = 1$. Then the ball $B(x_q^r, \varphi_j(\mathcal{T}_{j,q}(x_q^r)))$ is a subset of $\mathcal{F}_{j,q}^q(\mathbf{x}^r)$ according to (27) for all $j$ and $q$. Hence $B(x_q^r, d^*) \subset \bigcap_{j\geq 2} \mathcal{F}_{j,q}^q(\mathbf{x}^r)$, where $d^* = \min_{j\geq 2} \varphi_j(\mathcal{T}_{j,q}(x_q^r))$ is the minimum distance to $\bigcup_{j\geq 2} \mathcal{O}_j(q)$. According to (23), $\mathcal{F}_{1,q}^q(\mathbf{x}^r)$ is tangent to $\partial\mathcal{O}_1(q)$ at a point $x^*$ such that $\varphi_1(\mathcal{T}_{1,q}(x_q^r)) = -\|x_q^r - x^*\|$ as shown in Figure 10. Since $\bar{\mathcal{O}}_1(q) \bigcap \bar{\mathcal{O}}_j(q) = \emptyset$ for $j \geq 2$, then $d^* > -\varphi_1(\mathcal{T}_{1,q}(x_q^r)) = \|x_q^r - x^*\|$, which implies that the set $\mathcal{F}_{1,q}^q(\mathbf{x}^r) \bigcap B(x_q^r, d^*)$ has a nonempty interior. Then $\bigcap_j \mathcal{F}_{j,q}^q(\mathbf{x}^r)$ has a nonempty interior. So $\mathcal{F}(\mathbf{x}^r) = \oplus_q(\bigcap_j \mathcal{F}_{j,q}^q(\mathbf{x}^r))$ has a nonempty interior where $\oplus$ means direct sum. ∎

*Remark* 3. Lemma 6 implies that $\mathcal{F}^{(0)}$ is nonempty for any $\mathbf{x}^{(0)}$. By Theorem 1, the algorithm converges to a local optimum for any $\mathbf{x}^{(0)} \in \mathbb{R}^n$ if all obstacles are convex and have disjoint closures.

**5.4. Performance and comparison.** The performance of Algorithm 1 will be illustrated through two examples, which will also be compared to the performance of existing nonconvex optimization methods, ITP and SQP. For simplicity, only convex

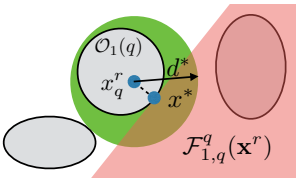



FIG. 10. *Existence of convex feasible set for an infeasible reference point.*

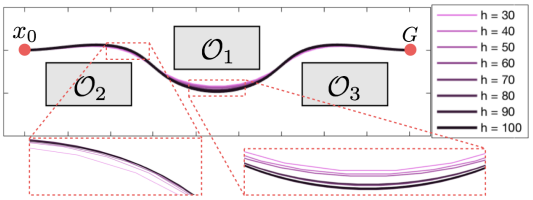



FIG. 11. *Scenario* 1 *and the optimal trajectories for different horizon* $h$.

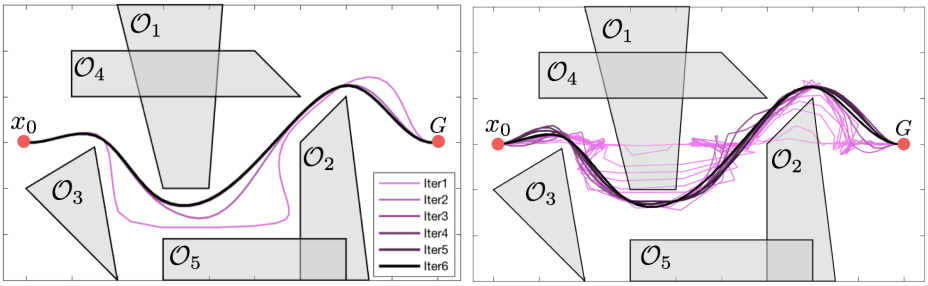
obstacles and convex boundaries are considered.[8] Algorithm 1 is implemented in both MATLAB and C++. The convex optimization problem (12) is solved using the interior-point-convex method in `quadprog` in MATLAB and ITP in Knitro [2] in C++. For comparison, (1) is also solved directly using ITP and SQP methods in `fmincon` [20] in MATLAB and in Knitro in C++. To create a fair comparison, the gradient and the hessian of the objective function $J$ and the optimal subgradients $\hat{\nabla}\phi_i$ of the constraint function $\phi_i$'s are also provided to ITP and SQP solvers.

In the examples, $x_0 = (0, 0)$ and $G = (9, 0)$. The planning horizon $h$ goes from 30 to 100. $t_s = (h + 1)^{-1}$. The cost function (21) penalizes the average acceleration along the trajectory, e.g., $Q = 0$ and $S = h^{-1}A^TA$. When $h \to \infty$, $J(\mathbf{x}) \to \int_0^1 \|\ddot{\mathbf{x}}\|^2 dt$, where $\mathbf{x} : [0, 1] \to \mathbb{R}^n$ is a continuous trajectory with $\mathbf{x}(0) = x_0$ and $\mathbf{x}(1) = G$. The initial reference $\mathbf{x}^{(0)}$ is chosen to be a straight line connecting $x_0$ and $G$ with equally sampled waypoints. In the first scenario, there are three disjoint convex obstacles as shown in Figure 11. In the second scenario, there are three disjoint infeasible sets, two of which contain concave corners. Then they are partitioned into five overlapping convex obstacles as shown in Figure 12. In the constraint, a distance margin of 0.25 to the obstacles is required.

The computation times under different solvers are listed in Table 1. The first column shows the horizon. In the first row, $h = 100$ in scenario 1 and $h = 60$ in scenario 2.[9] In the remaining rows, $h$ is the same in the two scenarios. In the second column, "-M" means the algorithm is run in MATLAB and "-C" means the algorithm is run in C++. Under each scenario, the first column shows the final cost. The second column is the total number of iterations. The third and fourth columns are the total computation time and the average computation time per iteration, respectively. (Only the entries that are less than 100ms are shown.) It does happen that the algorithms find different local optima, though CFS-M and CFS-C always find the same solution.

[8] The nonconvex obstacles or boundaries can either be partitioned into several convex components or be replaced with their convex envelops. Moreover, in practice, obstacles are measured by point clouds. The geometric information is extracted by taking the convex hull of the points. Hence it automatically partitions the obstacles into several convex polytopes.

[9] In the C++ solver, the constraint is limited by 300. Hence when there are five obstacles, the maximum allowed $h$ is 60.




(a) Trajectories in CFS-C. (b) Trajectories in ITP-C.

FIG. 12. *Scenario* 2 *and the trajectories before convergence for* $h = 50$.

TABLE 1
*Comparison among different algorithms.*

| | | Scenario 1 | | | | Scenario 2 | | | |
|---|---|---|---|---|---|---|---|---|---|
| $h$ | Method | Cost | Iter | Time | dT | Cost | Iter | Time | dT |
| 100 or 60 | SQP-M | 1358.9 | 239 | 140.5s | - | 5385.5 | 198 | 60.1s | - |
| | ITP-M | 1358.9 | 470 | 50.6s | - | 5349.6 | 320 | 18.0s | 56.3ms |
| | CFS-M | 1358.9 | 18 | 1.8s | 98.8ms | 5413.2 | 6 | 344.9ms | 57.5ms |
| | SQP-C | 1347.6 | 123 | 47.3s | - | 5489.6 | 52 | 10.2s | - |
| | ITP-C | 1341.7 | 306 | 2.9s | 9.5ms | 5349.6 | 123 | 595.0ms | 4.8ms |
| | CFS-C | 1358.9 | 18 | **74.4ms** | **4.1ms** | 5413.2 | 6 | **27.3ms** | **4.6ms** |
| 50 | SQP-M | 2299.5 | 110 | 25.8s | - | 5539.6 | 164 | 40.0s | - |
| | ITP-M | 1308.3 | 187 | 8.8s | 47.1ms | 5372.8 | 268 | 11.3s | 42.2ms |
| | CFS-M | 1458.2 | 8 | 212.1ms | 26.5ms | 5394.2 | 5 | 186.1s | 37.2ms |
| | SQP-C | 1308.3 | 52 | 3.4s | 65.4ms | 5682.0 | 62 | 5.7s | 91.9ms |
| | ITP-C | 1275.1 | 131 | 390ms | 3.0ms | 5555.8 | 96 | 495.6ms | 5.2ms |
| | CFS-C | 1458.2 | 8 | **23.7ms** | **3.0ms** | 5394.2 | 5 | **21.0ms** | **4.2ms** |
| 40 | SQP-M | 3391.6 | 97 | 15.6s | - | 5318.1 | 127 | 23.2s | - |
| | ITP-M | 1317.0 | 150 | 5.2s | 34.7ms | 5549.8 | 156 | 5.6s | 35.9ms |
| | CFS-M | 1317.0 | 8 | 172.6ms | 21.6ms | 5399.2 | 6 | 171.9 | 28.7ms |
| | SQP-C | 1317.0 | 40 | 1.5s | 37.5ms | 5568.8 | 69 | 2.8s | 40.6ms |
| | ITP-C | 1170.5 | 102 | 240.5ms | 2.4ms | 5399.2 | 91 | 290.4ms | 3.2ms |
| | CFS-C | 1317.0 | 8 | **16.5ms** | **2.1ms** | 5399.2 | 6 | **19.0ms** | **3.2ms** |
| 30 | SQP-M | 1039.2 | 106 | 8.4s | 79.2ms | 5075.8 | 90 | 11.0s | - |
| | ITP-M | 1039.2 | 109 | 2.8s | 25.7ms | 5162.4 | 127 | 3.2s | 25.2ms |
| | CFS-M | 1039.2 | 12 | 208.2ms | 17.3ms | 5167.3 | 5 | 110.6ms | 22.1ms |
| | SQP-C | 1453.3 | 27 | 379.1ms | 14.0ms | 5444.3 | 42 | 1.5s | 35.7ms |
| | ITP-C | 1039.2 | 59 | 118.5ms | 2.0ms | 5320.8 | 67 | 125.5ms | 1.9ms |
| | CFS-C | 1039.2 | 12 | **19.0ms** | **1.6ms** | 5167.3 | 5 | **12.2ms** | **2.4ms** |

In terms of computation time, Algorithm 1 always outperforms ITP and SQP, since it requires less time per iteration and fewer iterations to converge. This is due to the fact that CFS does not require additional line search after solving (12) as is needed in ITP and SQP, hence saving time during each iteration. CFS requires fewer iterations to converge since it can take unconstrained step length $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|$ in the convex feasible set as will be shown later. Moreover, Algorithm 1 scales much better than ITP and SQP, as the computation time and time per iteration in CFS-C go up almost linearly with respect to $h$ (or the number of variables).

The computation time of CFS consists of two parts: (1) the processing time, i.e., the time to compute $\mathcal{F}$, and (2) the optimization time, i.e., the time to solve (12). As shown in Figure 13, the two parts grow with $h$. In MATLAB, the processing time dominates, while the optimization time dominates in C++.

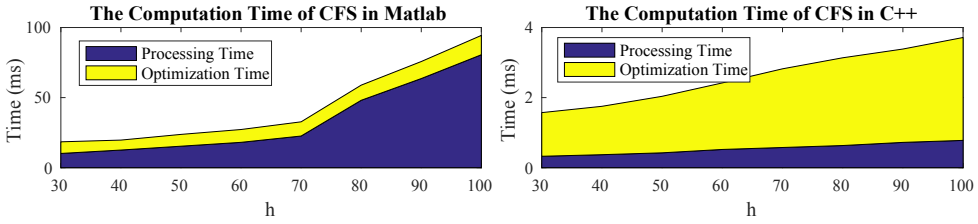



FIG. 13. *The decomposed time per iteration using Algorithm* 1 *in scenario* 1.

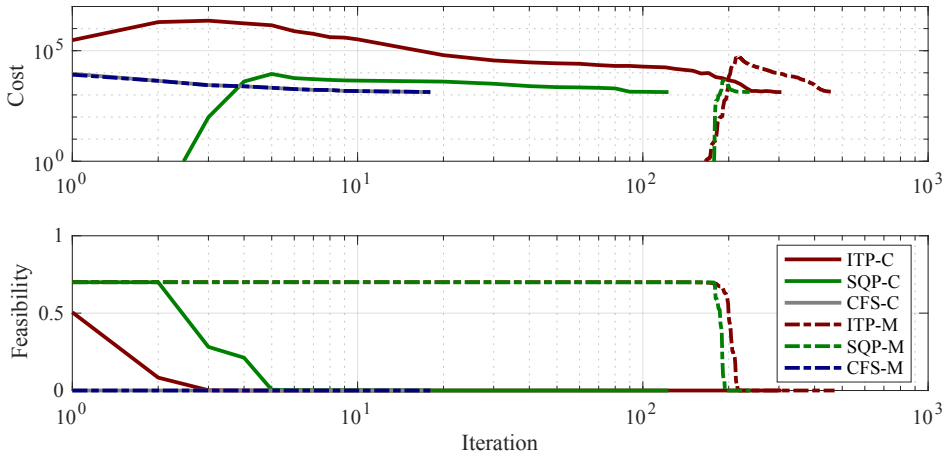



FIG. 14. *The run time statistics in scenario* 1.

To better illustrate the advantage of Algorithm 1, the runtime statistics across all methods when $h = 100$ in the first scenario are shown in Figure 14. The first log-log figure shows the cost $J(\mathbf{x}^{(k)})$ versus iteration $k$, while the second semi-log figure shows the feasibility error $\max\{0, -\phi_1(\mathbf{x}^{(k)}), \ldots, -\phi_N(\mathbf{x}^{(k)})\}$ versus $k$. At the beginning, the cost $J = 0$ and the feasibility error is 0.75. In CFS-C and CFS-M, $\mathbf{x}^{(k)}$ becomes feasible at the first iteration while the cost $J(\mathbf{x}^{(k)})$ jumps up. In the following iterations, the cost goes down and converges to the optimal value. In ITP-C, the problem becomes feasible at the third iteration. In SQP-C, it becomes feasible at the fifth iteration. In order to make the problem feasible, the cost jumps much higher in ITP-C than in CFS-C. Once the problem is feasible, it also takes more iterations for ITP-C and SQP-C to converge compared to CFS-C. On the other hand, ITP-M and SQP-M have very small step lengths in the beginning. The problem only becomes feasible after 100 iterations. But once the problem is feasible, the performance of ITP-M and SQP-M is similar to that of ITP-C and SQP-C. Note that the cost below 1 is not shown in the figure.

The optimal trajectories computed by Algorithm 1 for different $h$ in the first scenario are shown in Figure 11. Those trajectories converge to a continuous trajectory when $h$ goes up. Figure 12 illustrates the trajectories before convergence in CFS-C and ITP-C in scenario 2 when $h = 50$. For ITP-C, the trajectories are shown every iteration in the first ten iterations and then every ten iterations in the remaining iterations. The step length in CFS-C is much larger than that in ITP-C, which explains why CFS requires fewer iterations to become feasible and fewer iterations to converge. The trajectories are feasible and smooth in every iteration in CFS. Hence in

case of emergencies, we can safely stop the iterations and get a good enough feasible trajectory before convergence.

With respect to the results, we conclude that Algorithm 1 is time-efficient, local-optimal, and scalable.

**6. Conclusion.** This paper introduced a fast algorithm for real time motion planning based on the convex feasible set. The CFS algorithm can handle problems that have a convex cost function and nonconvex constraints which are usually encountered in robot motion planning. By computing a convex feasible set within the nonconvex constraints, the nonconvex optimization problem is transformed into a convex optimization. Then by iteration, we can efficiently eliminate the error introduced by the convexification. It is proved in the paper that the proposed algorithm is feasible and stable. Moreover, it can converge to a local optimum if either the initial reference satisfies certain conditions or the constraints satisfy certain conditions. The performance of CFS is compared to that of ITP and SQP. It is shown that CFS reaches local optima faster than ITP and SQP, and hence is better suited for real time applications. In the future, methods for computing the convex feasible sets for infeasible references in complicated environments will be explored.

REFERENCES

[1] B. Açikmeşe, J. M. Carson, and L. Blackmore, *Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem*, IEEE Trans. Control Syst. Technol., 21 (2013), pp. 2104–2113.

[2] R. H. Byrd, J. Nocedal, and R. A. Waltz, *Knitro: An Integrated Package for Nonlinear Optimization*, Springer, New York, 2006, pp. 35–59.

[3] C. Chen, M. Rickert, and A. Knoll, *Path planning with orientation-aware space exploration guided heuristic search for autonomous parking and maneuvering*, in Proceedings of the IEEE Intelligent Vehicles Symposium (IV), 2015, pp. 1148–1153.

[4] F. H. Clarke, *Generalized gradients and applications*, Trans. Amer. Math. Soc., 205 (1975), pp. 247–262.

[5] A. Colesanti and D. Hug, *Hessian measures of semi-convex functions and applications to support measures of convex bodies*, Manuscripta Math., 101 (2000), pp. 209–238.

[6] G. Eichfelder and J. Povh, *On the set-semidefinite representation of nonconvex quadratic programs over arbitrary feasible sets*, Optim. Lett., 7 (2013), pp. 1373–1386.

[7] G. B. Folland, *Real Analysis: Modern Techniques and Their Applications*, John Wiley and Sons, New York, 2013.

[8] E. Frazzoli, M. A. Dahleh, and E. Feron, *Real-time motion planning for agile autonomous vehicles*, J. Guid. Control Dyn., 25 (2002), pp. 116–129.

[9] M. W. Harris and B. Açikmeşe, *Lossless convexification of non-convex optimal control problems for state constrained linear systems*, Automatica, 50 (2014), pp. 2304–2311.

[10] T. M. Howard, C. J. Green, and A. Kelly, *Receding Horizon Model-Predictive Control for Mobile Robot Navigation of Intricate Paths*, in Field and Service Robotics, Springer, New York, 2010, pp. 69–78.

[11] T. A. Johansen, T. I. Fossen, and S. P. Berge, *Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming*, IEEE Trans. Control Syst. Technol., 12 (2004), pp. 211–216.

[12] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, *Real-time motion planning with applications to autonomous urban driving*, IEEE Trans. Control Syst. Technol., 17 (2009), pp. 1105–1118.

[13] J.-C. Latombe, *Robot Motion Planning*, Vol. 124, Springer, New York, 2012.

[14] C. Liu, C.-Y. Lin, Y. Wang, and M. Tomizuka, *Convex feasible set algorithm for constrained trajectory smoothing*, in Proceedings of the American Control Conference (ACC), IEEE, 2017, pp. 4177–4182.

[15] C. Liu and M. Tomizuka, *Algorithmic safety measures for intelligent industrial co-robots*, in Proceedings of the International Conference on Robotics and Automation (ICRA), IEEE, 2016, pp. 3095–3102.

[16] C. Liu and M. Tomizuka, *Enabling safe freeway driving for automated vehicles*, in Proceedings of the American Control Conference (ACC), IEEE, 2016, pp. 3461–3467.
[17] C. Liu and M. Tomizuka, *Real time trajectory optimization for nonlinear robotic systems: Relaxation and convexification*, Systems Control Lett., 108 (2017), pp. 56–63.
[18] X. Liu, *Autonomous Trajectory Planning by Convex Optimization*, Ph.D. thesis, Iowa State University, 2013.
[19] X. Liu and P. Lu, *Solving nonconvex optimal control problems by convex optimization*, J. Guid. Control Dyn., 37 (2014), pp. 750–765.
[20] *MathWorks, Optimization Toolbox, Constrained Optimization, Fmincon*, https://www.mathworks.com/help/optim/ug/fmincon.html.
[21] A. Matveev, M. Hoy, and A. Savkin, *A method for reactive navigation of nonholonomic under-actuated robots in maze-like environments*, Automatica, 49 (2013), pp. 1268–1274.
[22] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, New York, 2006.
[23] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, *CHOMP: Gradient optimization techniques for efficient motion planning*, in Proceedings of the International Conference on Robotics and Automation (ICRA), IEEE, 2009, pp. 489–494.
[24] A. V. Savkin, A. S. Matveev, M. Hoy, and C. Wang, *Safe Robot Navigation Among Moving and Steady Obstacles*, Butterworth-Heinemann, Oxford, 2015.
[25] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, *Finding locally optimal, collision-free trajectories with sequential convex optimization*, in Proceedings of the Robotics: Science and Systems, Vol. 9, 2013, pp. 1–10.
[26] P. Spellucci, *A new technique for inconsistent QP problems in the SQP method*, Math. Methods Oper. Res., 47 (1998), pp. 355–400.
[27] R. G. Strongin and Y. D. Sergeyev, *Global Optimization with Nonconvex Constraints: Sequential and Parallel Algorithms*, Vol. 45, Springer, New York, 2013.
[28] M. Tawarmalani and N. V. Sahinidis, *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*, Vol. 65, Springer, New York, 2002.
[29] K. Tone, *Revisions of constraint approximations in the successive QP method for nonlinear programming problems*, Math. Program., 26 (1983), pp. 144–152.
[30] Wayne State University, *Every convex function is locally Lipschitz*, Amer. Math. Monthly, 79 (1972), pp. 1121–1124.
[31] J. Van Den Berg, P. Abbeel, and K. Goldberg, *LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information*, Int. J. Robot. Res., 30 (2011), pp. 895–913.
[32] R. J. Vanderbei and D. F. Shanno, *An interior-point algorithm for nonconvex nonlinear programming*, Comput. Optim. Appl., 13 (1999), pp. 231–252.
[33] Z. Zhu, E. Schmerling, and M. Pavone, *A convex optimization approach to smooth trajectories for motion planning with car-like robots*, in Proceedings of the IEEE Conference on Decision and Control (CDC), IEEE, 2015, pp. 835–842.