SIAM J. CONTROL OPTIM.
Vol. 53, No. 2, pp. 712–744

OPTIMAL CONTROL WITH BUDGET CONSTRAINTS AND RESETS*

R. TAKEI†, W. CHEN‡, Z. CLAWSON§, S. KIROV¶, AND A. VLADIMIRSKY‖

Abstract. We consider both discrete and continuous control problems constrained by a fixed budget of some resource, which may be renewed upon entering a preferred subset of the state space. In the discrete case, we consider deterministic shortest path problems on graphs with a full budget reset in all preferred nodes. In the continuous case, we derive augmented PDEs of optimal control, which are then solved numerically on the extended state space with a full/instantaneous budget reset on the preferred subset. We introduce an iterative algorithm for solving these problems efficiently. The method's performance is demonstrated on a range of computational examples, including optimal path planning with constraints on prolonged visibility by a static enemy observer.

Key words. hybrid systems, optimal control, reset-renewable resources, contiguous visibility, integral constraints, Hamilton–Jacobi, discontinuous viscosity solution

AMS subject classifications. 90C29, 49L20, 49L25, 49N90, 65N22, 65K05, 34A38

DOI. 10.1137/110853182

1. Introduction.

Dynamic programming provides a convenient framework for finding provably "optimal" strategies to control both discrete and continuous systems. The optimality is usually defined with respect to a single criterion or cost (e.g., money, fuel, or time needed to implement each particular control). Given a set of possible system configurations Ω, the *value function* is defined as the cost of such optimal control for each starting position $\boldsymbol{x} \in \Omega$, and the dynamic programming equations are then solved to recover this value function.

Realistic applications usually involve several different criteria for evaluating control strategies and the notion of "optimality" becomes more complicated. One natural approach is to focus on a single "primary" cost to be optimized, while treating all other ("secondary") costs as constraints. A typical application of this type is to find the fastest path to the target subject to constraints on the maximum use of energy along that path. Another possible constraint is on the maximum total amount of time that a robot can be visible to an unfriendly observer while moving to a target. Kumar and Vladimirsky have recently introduced efficient numerical methods for these and similar constrained-optimal control problems in continuous domains [27]. We will refer to such problems as "budget-constrained" since the original state space will have

*Received by the editors October 27, 2011; accepted for publication (in revised form) July 7, 2014; published electronically March 19, 2015. The research of the second, third, and fourth authors was supported in part by the National Science Foundation through the Research Experiences for Undergraduates Program at Cornell.

http://www.siam.org/journals/sicon/53-2/85318.html

†Department of Electrical Engineering and Computer Sciences, UC Berkeley, CA 94720 (rrtakei@gmail.com). This author's research was supported in part by ONR grants N0014-03-1-0071, N00014-07-1-0810, N00014-08-1-1119, DOE grant DE-FG02-05ER25710, an ARO MURI through Rice University, and the CHASE MURI grant 556016.

‡Department of Mathematics, Cornell University, Ithaca, NY 14853 (wc363@cornell.edu).

§Center for Applied Mathematics, Cornell University, Ithaca, NY 14853 (zc227@cornell.edu).

¶Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213 (skirov@andrew.cmu.edu).

‖Department of Mathematics, Center for Applied Mathematics, Cornell University, Ithaca, NY 14853 (vlad@math.cornell.edu). This author's research was supported in part by the National Science Foundation grant DMS-1016150.

to be expanded to keep track of the remaining "resource-budget" to satisfy the constraint. (For example, two different starting energy-budgets may well result in very different "constrained time-optimal" paths, even when the starting physical position is the same.)

This state space expansion leads to a significantly larger system of augmented dynamic programming equations or a higher-dimensional domain in the continuous case. Thus, the computational efficiency of numerical methods becomes particularly important. In many applications it is natural to assume that the resource budgets are nonrenewable and that every change in the system state results in an immediate budget decrease. This introduces a natural "causal ordering" on the expanded state space: the knowledge of energy-constrained time-optimal controls for the starting energy-budget $\boldsymbol{b}_1$ can be used in computing such constrained-optimal controls for a higher budget $\boldsymbol{b}_2 > \boldsymbol{b}_1$. The efficiency of methods introduced in [27] hinges on this explicit causality.

In the current paper we focus on a more general situation, where the budget can be *instantaneously reset* to its maximum value by visiting a special part of the state space $\mathcal{S} \subset \Omega$ and the budget is decreasing when moving through the rest of the state space $\mathcal{U} = \Omega \backslash \mathcal{S}$. If the limited resource is fuel, $\mathcal{S}$ can be thought of as a discrete set of gas stations. On the other hand, if the secondary cost is the vehicle's exposure to an unfriendly observer, the $\mathcal{S}$ can be interpreted as a "safe" part of the domain protected from observation, and the constraint is on the maximum *contiguous* period of time that the system is allowed to spend in an "unsafe" (observable) set $\mathcal{U}$. To the best of our knowledge, our formulation of continuous versions of such problems is new and no efficient methods for these were previously available. We show that such "budget-resets" result in a much more subtle implicit causality in the expanded state space. Nevertheless, under mild technical assumptions, noniterative methods are available for deterministic (section 2) and even for certain stochastic budget-reset problems on graphs. Methods for the latter are discussed in an extended version of this manuscript [37]. In the continuous case, this problem is described by a controlled hybrid system (section 3), whose value function is a discontinuous viscosity solution of two different Hamilton–Jacobi PDEs posed on $\mathcal{S}$ and $\mathcal{U} \times \mathcal{B}$, where $\mathcal{B}$ is the set of possible budget levels available to satisfy secondary constraints. The characteristics of these PDEs coincide with the constrained-optimal trajectories and define the direction of information flow in the continuous state space. Unfortunately, the most natural semi-Lagrangian discretization of these PDEs is no longer causal, making iterative numerical methods unavoidable (section 4).

The key contributions of this paper are the study of properties of the value functions for budget reset problems (section 3.3) and the fast iterative algorithm for approximating such value functions (sections 4.2–4.4). Our general approach is equally efficient even in the presence of strong inhomogeneities and anisotropies in costs and/or dynamics of the system. In section 5, we provide numerical evidence of the method's convergence and illustrate the key properties on several optimal control examples,[1] including prolonged-visibility-avoidance problems.

We note that for budget-reset problems, finding constrained-optimal controls is significantly harder than just identifying the "reachable" subset of Ω; i.e., the set of states from which it's possible to steer the system to the target without violating the constraints, provided one starts with the maximum budget. A more efficient algorithm

[1] MATLAB/C++ codes used to produce all experimental results in this paper are available from http://www.math.cornell.edu/∼vlad/papers/b_reset/.

for the latter problem is included in [37]. We conclude by discussing the limitations of our approach and the future work in section 6.

2. Deterministic SP on graphs with renewable resources. The classical problem of finding the shortest path (SP) in a directed graph with nonnegative arc-lengths is among those most exhaustively studied. Fast iterative ("label-correcting") and fast noniterative ("label-setting") methods for it are widely used in a variety of applications.

Consider a directed graph on a set of nodes $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_M, \boldsymbol{x}_{M+1} = \boldsymbol{t}\}$, where the last node $\boldsymbol{t}$ is the target. For each node $\boldsymbol{x}_i \in X$, there is a set of immediate neighbors $N_i = N(\boldsymbol{x}_i) \subset X \backslash x_i$ and for each neighbor $\boldsymbol{x}_j \in N_i$ there is a known transition cost $C_{ij} = C(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$. For convenience, we will take $C_{ij} = +\infty$ if $\boldsymbol{x}_j \notin N_i$. We will further assume that our graph is relatively sparse; i.e., $\max_i |N_i| \leq \kappa \ll M$. If $\boldsymbol{y} = (\boldsymbol{y}_0 = \boldsymbol{x}_i, \boldsymbol{y}_1, \dots, \boldsymbol{y}_r = \boldsymbol{t})$ is a path from $\boldsymbol{x}_i$ to the target, its total cost is defined as $\mathcal{J}(\boldsymbol{y}) = \sum_{k=0}^{r-1} C(\boldsymbol{y}_k, \boldsymbol{y}_{k+1})$. The value function $U_i = U(\boldsymbol{x}_i)$ is defined as the cost along an optimal path. Clearly $U_{\boldsymbol{t}} = 0$ and for all other nodes the Bellman optimality principle yields a system of M nonlinear coupled equations:

$$U_i = \min_{\boldsymbol{x}_j \in N_i} \{C_{ij} + U_j\}; \qquad i \in \mathcal{I} = \{1, \dots, M\}. \tag{2.1}$$

(This definition makes $U_i = +\infty$, whenever there is no path from $\boldsymbol{x}_i$ to $\boldsymbol{t}$, including the cases where $N_i = \emptyset$. Throughout this paper, we will use the convention that the minimum over an empty set is $+\infty$.) Dijkstra's classical method [18] can be used to solve the system (2.1) noniteratively in $O(M \log M)$ operations. A detailed discussion of this and other label-setting and label-correcting algorithms can be found in standard references; see, e.g., [1, 8].

In many applications, a natural extension of this problem requires keeping track of several different types of cost (e.g., money, fuel, time) associated with each transition. The goal is then either to find all Pareto optimal paths or to treat one criterion/cost as primary (to be minimized) and all others as secondary, providing the constraints to restrict the set of allowable paths. (For example, what is *the quickest* path given the current amount of fuel in our tank?) A number of algorithms for such multiobjective dynamic programming are also well-known [24, 29, 25].

One natural approach to bicriterion problems involves finding a simple, single-criterion optimal path but in an expanded graph with the node set $\widehat{X} = X \times \mathcal{B}$. We begin with a similar technique adopted to our "constrained resources" scenario. Our description emphasizes the causal properties of the model, also reflected by the numerical methods for the continuous case in section 4. We assume the secondary resource-cost for each transition is specified as $c_{ij} = c(\boldsymbol{x}_i, \boldsymbol{x}_j)$, whereas the primary cost of that transition will be still denoted as C_{ij}. For simplicity we assume that the secondary costs are conveniently quantized; e.g., $c_{ij} \in \mathbb{Z}$.

We will use $B > 0$ to denote the maximal allowable budget and $\mathcal{B} = \{0, \dots, B\}$ to denote the set of allowable budget levels. In the extended graph, a node $\boldsymbol{x}_i^b \in \widehat{X}$ corresponds to a location $\boldsymbol{x}_i \in X$ with the resource budget b. The use of resources occurs when $c_{ij} > 0$, and the renewal/recharge of resources corresponds to $c_{ij} < 0$. If we are starting from $\boldsymbol{x}_i$ with a secondary-budget b and decide to move to $\boldsymbol{x}_j \in N_i$, in the expanded graph this becomes a transition from $\boldsymbol{x}_i^b$ to $\boldsymbol{x}_j^{b-c_{ij}}$.

We now define the value function $W_i^b = W(\boldsymbol{x}_i^b)$ as the minimum accumulated primary-cost from $\boldsymbol{x}_i$ to $\boldsymbol{t}$, but minimizing only among the paths along which the

budget always remains in $\mathcal{B}$:

$$W_i^b = \min_{\boldsymbol{y} \in Y^b(\boldsymbol{x}_i)} \mathcal{J}(\boldsymbol{y}),$$

where $Y^b(\boldsymbol{x}_i)$ is the set of "b-feasible paths" (i.e., those that can be traversed if starting from $\boldsymbol{x}_i$ with resource budget b).

Feasible paths should clearly include only those along which the resource budget remains nonnegative. But since we are allowing for secondary costs c_{ij} of arbitrary sign, this requires a choice between two different "upper budget bound" interpretations:

(1) Any attempt to achieve a budget higher than B makes a path infeasible; i.e., since we are starting from $\boldsymbol{x}_i$ with budget $b \in \mathcal{B}$,

$$Y^b(\boldsymbol{x}_i) = \left\{ \boldsymbol{y} = (\boldsymbol{y}_0 = \boldsymbol{x}_i, \ldots, \boldsymbol{y}_r = \boldsymbol{t}) \mid 0 \le \left(b - \sum_{k=0}^{s-1} c(\boldsymbol{y}_k, \boldsymbol{y}_{k+1}) \right) \le B, \quad \forall s \le r \right\}.$$

In this case, the optimality principle would yield a system of equations on the expanded graph:

$$W_i^b = \min_{\boldsymbol{x}_j \in N_i} \left\{ C_{ij} + W_j^{b - c_{ij}} \right\}; \qquad \forall b \in \mathcal{B}; \quad \forall i \in \mathcal{I};$$

with the following "boundary conditions":

$$W_{\boldsymbol{t}}^b = 0; \quad \forall b \in \mathcal{B}; \qquad W_i^b = +\infty, \quad \forall b \notin \mathcal{B}, \quad \forall \boldsymbol{x}_i \in X.$$

In practice, the latter condition can be omitted if we minimize over $N_i^b = \{\boldsymbol{x}_j \in N_i \mid (b - c_{ij}) \in \mathcal{B}\}$ instead of N_i.

(2) An interpretation more suitable for our purposes is to assume that any resources in excess of B are simply not accumulated (or are immediately lost), but the path remains allowable. If we define a left-associative operation $\alpha \ominus \beta = \min(\alpha - \beta, B)$, then

$$Y^b(\boldsymbol{x}_i) = \left\{ \boldsymbol{y} = (\boldsymbol{y}_0 = \boldsymbol{x}_i, \ldots, \boldsymbol{y}_r = \boldsymbol{t}) \mid \left(b \ominus c(\boldsymbol{y}_0, \boldsymbol{y}_1) \ominus \cdots \ominus c(\boldsymbol{y}_{s-1}, \boldsymbol{y}_s) \right) \ge 0, \quad \forall s \le r \right\},$$

and the optimality principle on the expanded graph can be expressed as follows:

$$W_i^b = \min_{\boldsymbol{x}_j \in N_i^b} \left\{ C_{ij} + W_j^{b \ominus c_{ij}} \right\}; \qquad \forall b \in \mathcal{B}; \quad \forall i \in \mathcal{I}, \tag{2.2}$$

with $W_{\boldsymbol{t}}^b = 0$ for all $b \in \mathcal{B}$ and $N_i^b = \{\boldsymbol{x}_j \in N_i \mid c_{ij} \le b\}$. Unlike the previous interpretation, this definition ensures that the value function is nonincreasing in b (since the set of feasible paths is nondecreasing as we increase the budget). Thus, we will also similarly interpret the "upper budget bound" in section 3.

Remark 1. It is natural to view the expanded graph as $(B+1)$ "b-slices" (i.e., copies of the original graph) stacked vertically, with the transitions between slices corresponding to c_{ij}'s. (Though, since the total costs of feasible paths do not depend on the budget remaining upon reaching the target, it is in fact unnecessary to have multiple copies of $\boldsymbol{t}$ in the expanded graph; see Figure 2.) We note that the signs of the secondary costs strongly influence the type of causality present in the system as well as the efficiency of the corresponding numerical methods. We consider three cases:

1. Since the primary costs are nonnegative, the system (2.2) can always be solved by Dijkstra's method on an expanded graph regardless of the signs of c_{ij}'s. The computational cost of this approach is $O(\hat{M}\log\hat{M})$, where $\hat{M} = M \times (B+1)$ is the number of nodes in the expanded graph (not counting the target). We will refer to this most general type of causality as *implicit*.

2. However, if c_{ij}'s are strictly positive, Dijkstra's method becomes unnecessary since the budget will strictly decrease along every path. In this case $W_i^0 = +\infty$ for all $\boldsymbol{x}_i \in X\backslash\{\boldsymbol{t}\}$ and each W_i^b depends only on nodes in the lower slices. We will refer to this situation as *explicitly causal* since we can compute the value function in $O(\hat{M})$ operations in a single sweep from the bottom to the top slice.

3. On the other hand, if the secondary costs are only nonnegative, the budget is nonincreasing along every path and we can use Dijkstra's method on one b-slice at a time (again, starting from $b = 0$ and advancing to $b = B$) instead of using it on the entire expanded graph at once. This *semi-implicit causality* results in the computational cost of $O(\hat{M}\log M)$.

Remark 2. The sign of secondary costs also determines whether the value function W_i^b actually depends on the maximum allowable resource level. For example, suppose the solution W_i^b was already computed for all $b \in \mathcal{B} = \{0,\dots,B\}$ and it is now necessary to solve the problem again, but for a wider range of resource budgets $\tilde{\mathcal{B}} = \{0,\dots,\tilde{B}\}$, where $\tilde{B} > B$. Assuming that $\tilde{W}$ solves the latter problem, a useful consequence of explicit and semi-implicit causality is the fact that $\tilde{W}_i^b = W_i^b$ for all $b \in \mathcal{B}$; thus, the computational costs of this extension are decreased by concentrating on the set of budgets $\{B+1,\dots,\tilde{B}\}$ only. Unfortunately, this convenient property does not generally hold for implicitly causal problems. (For example, when refueling is allowed, two cars with different capacity of gas tanks might have different optimal driving directions even if they are starting out with the same amount of gas.)

2.1. Safe/unsafe splitting without budget resets. The framework presented so far is suitable for fairly general resource expenditure/accumulation models. In this paper, we are primarily interested in a smaller class of problems, where the state space is split into "safe" and "unsafe" components (i.e., $X\backslash\{\boldsymbol{t}\} = \mathcal{S}\cup\mathcal{U}$) with the secondary cost not accrued (i.e., the constrained resource not used at all) while traveling through $\mathcal{S}$. A simple example of this is provided in Figure 1.

In the absence of "budget resets" this safe/unsafe splitting is modeled by simply setting $c_{ij} = 0$ whenever $\boldsymbol{x}_i \in \mathcal{S}$, yielding semi-implicit causality; see Figures 2 and 3. To make this example intuitive, we have selected the primary costs C_{ij} to be such that the "primary-optimal" path (described by U; see (2.1)) always proceeds from $\boldsymbol{x}_i$ to $\boldsymbol{x}_{i+1}$, etc. However, the budget limitations can make such primary-optimal paths infeasible. For example, when $B = 3$ and no resets are allowed, the best feasible path from $\boldsymbol{x}_4$ is $(\boldsymbol{x}_4, \boldsymbol{x}_5, \boldsymbol{x}_8, \boldsymbol{t})$, resulting in $W_4^3 = 6 > 5 = U_4$; see Figure 3. Still, all constrained-optimal paths travel either on the same b slice or downward, resulting in a semi-implicit causality.

Since for large B the expanded graph is significantly bigger than the original one, it is useful to consider "fast" techniques for pruning $\widehat{X}$. We note that the $W_i^0 = \infty$ for all $\boldsymbol{x}_i \in \mathcal{U}$ and so the "0-budget" copies of $\mathcal{U}$ nodes can always be safely removed from the extended graph. A more careful argument can be used to reduce the computational domain much further.

To begin, we describe the resource-optimal paths by defining another value function $V_i = V(\boldsymbol{x}_i)$ to be the minimum resource budget b needed to reach the target from $\boldsymbol{x}_i$. For the described model, $c_{ij} \geq 0$ and it is easy to show that this new "resource

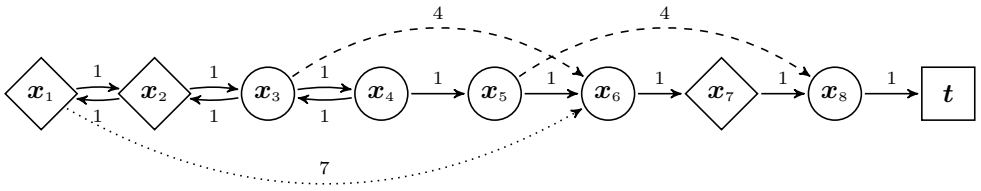


FIG. 1. *Diamond-shaped nodes are in $\mathcal{S}$ and circle-shaped ones are in $\mathcal{U}$. The primary costs C_{ij} are specified for each link, while the secondary costs are $c_{ij} = 1$ if $\boldsymbol{x}_i \in \mathcal{U}$ and $c_{ij} = 0$ if $\boldsymbol{x}_i \in \mathcal{S}$. The arrow types (solid, dashed, or dotted) also correspond to different values of primary transition costs. To build a concrete illustration, we will assume that $B = 3$ is the maximum number of "unsafe" (circle-shaped) nodes we can go through.*

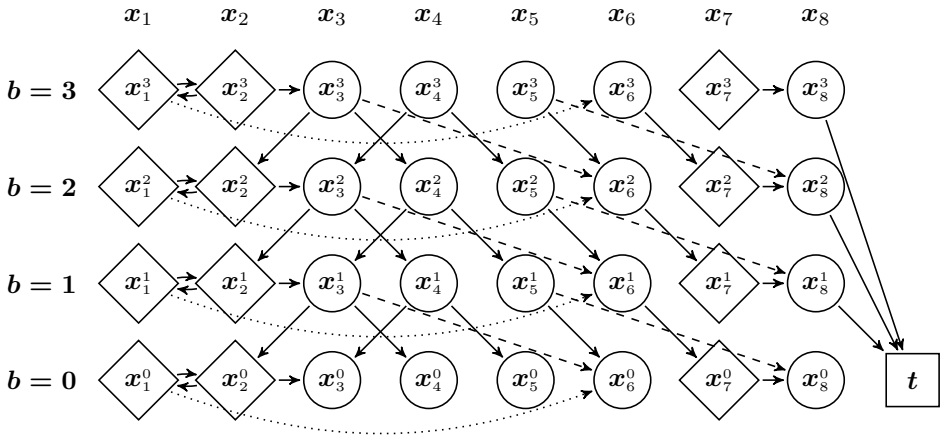


FIG. 2. *Budget-constrained shortest path (no "resets"). For every node except for the target, the superscript denotes* the remaining unsafe-node budget *at that node. Transition on the same level from a safe node, down to the next level if going from an unsafe one. The primary transition costs are indicated by the type of arrow; see Figure* 1.

value function" would have to satisfy

$$V_i = \min_{\boldsymbol{x}_j \in N_i} \{c_{ij} + V_j\}, \qquad i \in \mathcal{I} = \{1, \dots, M\}. \tag{2.3}$$

We will further define $\tilde{V}_i$ as the minimum resource budget sufficient to follow a primary-optimal path from $\boldsymbol{x}_i$. The equation for $\tilde{V}_i$ is similar to (2.3), but with the minimum taken over the set of optimal transitions in (2.1) (instead of minimizing over the entire N_i). Analogously, we can define $\tilde{U}_i$ to be the primary cost along the resource-optimal trajectories.

Definitions of these four functions are summarized in the caption of Table 1. The left side of this table shows the values of $U, \tilde{V}, V$, and $\tilde{U}$ for the example presented in Figure 1. We note the following:

(1) $W_i^b = \tilde{U}_i$ for $b = V_i$. We will refer to the set of such $\boldsymbol{x}_i^b$ nodes as the *Minimum Feasible Level* since $W_i^\beta = \infty$ for any $\beta < V_i$.

(2) $W_i^b = U_i$ for any $b \geq \tilde{V}_i$, since the primary-optimal trajectories are feasible there.

These two observations can be used to strongly reduce the extended graph on

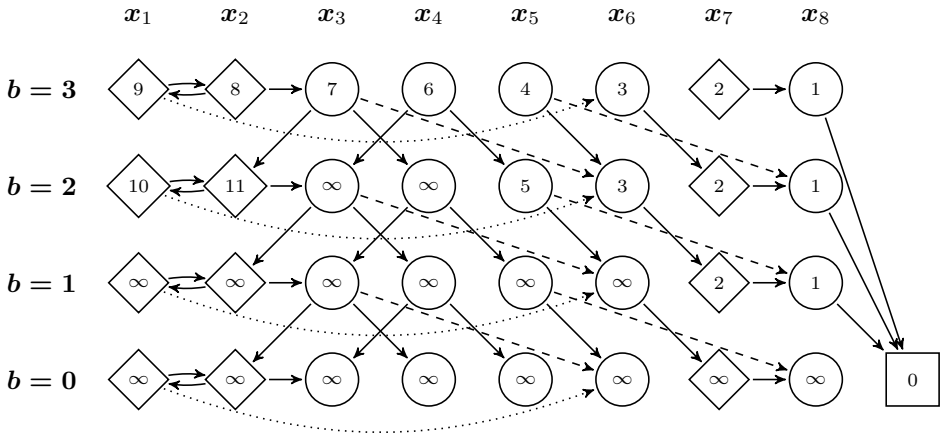


FIG. 3. *Semi-implicit causality: run Dijkstra's algorithm* slice-by-slice *on the expanded graph (from bottom to top). The number inside each node represents the minimum cost from* $\boldsymbol{x}_i^b$ *to the target. (If there are no "safe cycles," the causality becomes explicit and there is no need for Dijkstra's.)*

which we solve the system (2.2). For example, in Figure 3, this pruning excludes all nodes except for $\{\boldsymbol{x}_1^3, \boldsymbol{x}_2^3\}$. In examples where $V_i \ll B \ll \tilde{V}_i$ for most nodes, this remaining subset of $\widehat{X}$ will generally be much larger, but the pruning is still worthwhile since $U, \tilde{V}, V$, and $\tilde{U}$ can be computed on a much smaller (original) graph. A similar procedure for the continuous case is described in section 4.4.

TABLE 1

Various "optimal" costs for the example in Figure 1 with $B = 3$. *Primary-optimal cost* (U), *resource-optimal cost* (V), *resource cost along primary-optimal path* ($\tilde{V}$), *and primary cost along resource-optimal path* ($\tilde{U}$) *for the no-reset and budget-reset problems. Note that for the latter problem we can also define* $\tilde{V}(\boldsymbol{x}_3) = \infty$ *since* $B = 3$.

Without budget resets.

	$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	$\boldsymbol{x}_3$	$\boldsymbol{x}_4$	$\boldsymbol{x}_5$	$\boldsymbol{x}_6$	$\boldsymbol{x}_7$	$\boldsymbol{x}_8$
U	8	7	6	5	4	3	2	1
$\tilde{V}$	5	5	5	4	3	2	1	1
V	2	2	3	3	2	2	1	1
$\tilde{U}$	10	11	7	6	5	3	2	1

With budget resets.

	$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	$\boldsymbol{x}_3$	$\boldsymbol{x}_4$	$\boldsymbol{x}_5$	$\boldsymbol{x}_6$	$\boldsymbol{x}_7$	$\boldsymbol{x}_8$
U	8	7	6	5	4	3	2	1
$\tilde{V}$	-	-	4	3	2	1	-	1
V	-	-	1	2	2	1	-	1
$\tilde{U}$	-	-	9	10	4	3	-	1

2.2. Safe/unsafe splitting with budget resets. Our main focus, however, is on problems where the budget is also fully reset upon entering $\mathcal{S}$. To model that, we can formally set $c_{ij} = -B < 0$, ensuring the transition from $\boldsymbol{x}_i^b$ to $\boldsymbol{x}_j^B$, whenever $\boldsymbol{x}_i \in \mathcal{S}$. Since we always have a maximum budget in $\mathcal{S}$, there is no need to maintain a copy of it in each b-slice. This results in a smaller expanded graph with $(|\mathcal{S}| + B|\mathcal{U}| + 1)$ nodes; see Figure 4. Unfortunately, the negative secondary costs make this problem implicitly causal, impacting the efficiency. Moreover, unlike the no-resets case, the projections of constrained-optimal trajectories onto the original graph may contain loops. For example, starting from $\boldsymbol{x}_4$ with the resource budget $b = 2$, the constrained-optimal trajectory is $(\boldsymbol{x}_4, \boldsymbol{x}_3, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_6, \boldsymbol{x}_7, \boldsymbol{x}_8, \boldsymbol{t})$ and the first two transitions are needed to restore the budget to the level sufficient for the rest of the path. Note that, if we were to re-solve the same problem with $B = 4$, the constrained-optimal path from the same starting location and budget would be quite different:

$(\boldsymbol{x}_4, \boldsymbol{x}_3, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_5, \boldsymbol{x}_6, \boldsymbol{x}_7, \boldsymbol{x}_8, \boldsymbol{t})$, resulting in a smaller $W_4^2 = 9$; see Remark 2.

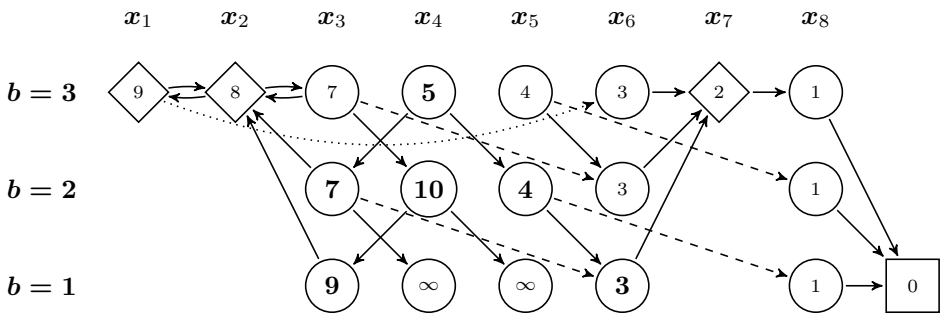


FIG. 4. *The problem with budget resets for $B = 3$. (The $b = 0$ slice is omitted simply because $W_i^0 = \infty$ for any $\boldsymbol{x}_i \in \mathcal{U}$.) Implicit/monotone causality: run Dijkstra's algorithm* on the entire expanded graph. *The number inside each node represents the minimum cost from $\boldsymbol{x}_i^b$ to the target. We show in bold font the values different from the "no resets" case.*

Unfortunately, the domain restriction/pruning techniques outlined in the previous subsection are no longer directly applicable. For the current example, the values of functions U, $\tilde{V}$, V, and $\tilde{U}$ are provided in the right half of Table 1, but their interpretations (and algorithms needed to compute them) are more subtle. For example, the function V_i is naturally interpreted as the minimum starting budget sufficient to reach $\boldsymbol{t}$ starting from $\boldsymbol{x}_i$. Thus, V_i is not defined on "safe" nodes (since on $\mathcal{S}$ the budget is always equal to B), and for $\boldsymbol{x}_i \in \mathcal{U}$ the value of V_i is B-dependent. If $\bar{b} = V_i$, then $Y^{\bar{b}}(\boldsymbol{x}_i) \neq \emptyset$ (i.e., there exists a $\bar{b}$-feasible path from $\boldsymbol{x}_i$), but $Y^b(\boldsymbol{x}_i) = \emptyset$ for all $b < \bar{b}$. If $\boldsymbol{y} = (\boldsymbol{y}_0 = \boldsymbol{x}_i, \ldots, \boldsymbol{y}_r = \boldsymbol{t}) \in Y^{\bar{b}}(\boldsymbol{x}_i)$ and s is the smallest index such that $\boldsymbol{y}_s \in \mathcal{S}$, then $\bar{b} = \sum_{k=0}^{s-1} c(\boldsymbol{y}_k, \boldsymbol{y}_{k+1})$.

As a result, we could also interpret V_i as the minimum "resource cost" of traveling from $\boldsymbol{x}_i$ through $\mathcal{U}$ to reach $\hat{\mathcal{S}} = \{\boldsymbol{x}_j \in \mathcal{S} \mid W_j^B < \infty\}$.

Unfortunately, this definition does not allow for efficient computations on the original graph, since $\hat{\mathcal{S}}$ is not known a priori.

If the values W_j^B were already known for all the "Safe" nodes $\boldsymbol{x}_j \in \mathcal{S}$, we could efficiently compute V_i for all $\boldsymbol{x}_i \in \mathcal{U}$ (thus restricting the computational domain); moreover, all W_i^b could then be computed in a single upward sweep (from the slice $b = 1$ to the slice $b = B$). This observation serves as a basis for the iterative method summarized in Algorithm 1. Intermediate stages of this algorithm applied to the above example are depicted in Figure 5.

Since the primary costs C_{ij}'s are positive, the value function can still be computed noniteratively—by using Dijkstra's method on the full expanded graph without attempting any domain restriction techniques. It is possible to find examples on which the advantages of restricting the domain outweigh the noniterative nature of Dijkstra's method. But our main reason for describing the iterative Algorithm 1 is its applicability to the problems of section 3, where Dijkstra-like methods are generally inapplicable.

3. Continuous problems with budget reset.

3.1. Continuous optimal control formulation. We begin with a brief review of the basic exit time optimal control problem in the absence of resource constraints. Given an equation for controlled motion in Ω, the objective is to characterize optimal paths (and their associated optimal costs) from every point in a domain Ω

Initialization:

$W_{\boldsymbol{t}} := 0$;
$W_j^B := \infty \qquad \forall\ \boldsymbol{x}_j \in \mathcal{S}$;
$W_i^b := \infty \qquad \forall\ \boldsymbol{x}_i \in \mathcal{U},\ b \in \mathcal{B}$.
Compute U_i for all $\boldsymbol{x}_i \in X$.

Main Loop:

repeat
  Phase I:
  Using the current W_j^B values to specify $\hat{\mathcal{S}}$, compute V_i and $\tilde{U}_i$ for each $\boldsymbol{x}_i \in \mathcal{U}$.
  foreach $b = 1, \ldots, B$ **do**
    foreach $\boldsymbol{x}_i \in \mathcal{U}$ **do**
      if $b \geq V_i$ **then**
        if $b = V_i$ **then**
          $W_i^b := \tilde{U}_i$;
        else
          if $W_i^{b-1} = U_i$ **then**
            $W_i^b := U_i$;
          else
            Compute W_i^b from equation (2.2).
          end
        end
      end
    end
  end
  Phase II:
  Run Dijkstra's method **on** $\mathcal{S}$ **only** to (re)compute W_j^B's for all $\boldsymbol{x}_j \in \mathcal{S}$.
until *all* W_j^B*'s stop changing*;

Algorithm 1: Iterative algorithm for the DSP problem with budget resets.

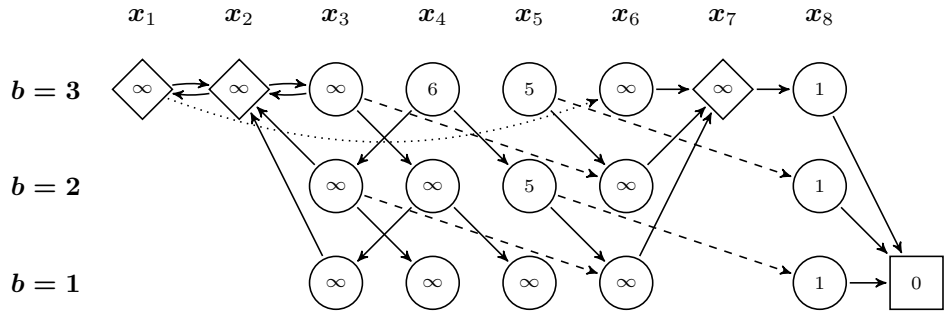
to the boundary $\partial\Omega$. The static Hamilton–Jacobi–Bellman (HJB) PDE discussed in this subsection is fairly standard and we focus only on the details important for the subsequent discussion. For a comprehensive treatment of more general resource-unconstrained problems, we refer interested readers to the monograph [4] and the references within.

Consider an open bounded set $\Omega \subset \boldsymbol{R}^n$. Let $A \in \boldsymbol{R}^n$ be a compact set of control values and $\mathcal{A}$ the set of admissible controls

$$\mathcal{A} = \{\text{measurable functions } \boldsymbol{a}\colon\ [0, +\infty) \to A\}. \tag{3.1}$$

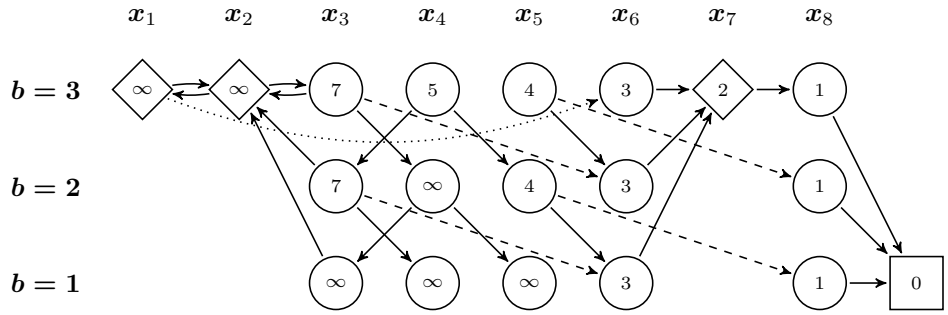
Initialize: $W_1^3 = W_2^3 = W_7^3 = \infty$.

Fix "Safe" and solve on "Unsafe":



Fix "Unsafe" and solve on "Safe": $W_7^3 = 2$.

Fix "Safe" and solve on "Unsafe":



Fix "Unsafe" and solve on "Safe": $W_1^3 = 9; \quad W_2^3 = 8.$

Fix "Safe" and solve on "Unsafe":

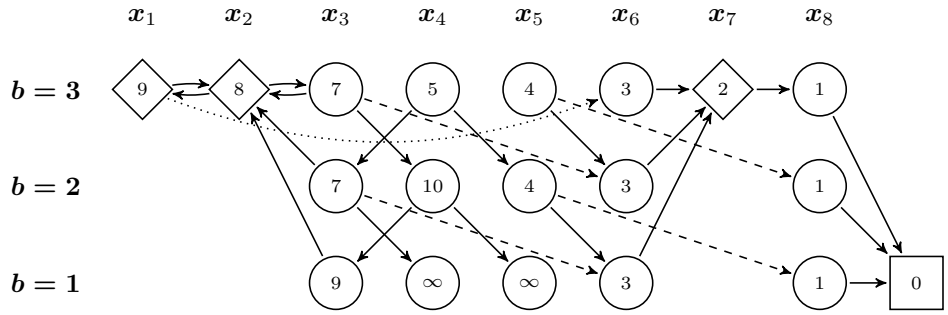


FIG. 5. *The problem with budget resets for* $B = 3$ *(specified in Figure* 4*) solved by an iterative Algorithm* 1*. Note that the MFL for* $\boldsymbol{x}_3$ *and* $\boldsymbol{x}_4$ *is not correctly determined until the last iteration.*

Let $\boldsymbol{y}\colon [0,\infty) \to \Omega$ represent the time-evolution of a particle state governed by the dynamical system

$$\begin{aligned} \frac{d\boldsymbol{y}}{dt}(t) = \dot{\boldsymbol{y}}(t) &= \boldsymbol{f}(\boldsymbol{y}(t), \boldsymbol{a}(t)), \\ \boldsymbol{y}(0) &= \boldsymbol{x}, \end{aligned} \tag{3.2}$$

where $\boldsymbol{f}\colon \Omega \times A \to \boldsymbol{R}^n$ represents the velocity. We refer to all $\boldsymbol{y}(\cdot)$ that satisfy (3.2) with admissible controls $\mathcal{A}$ as *admissible paths*. For notational simplicity, we have suppressed explicitly writing the dependence of $\boldsymbol{y}(\cdot)$ on the control $\boldsymbol{a} \in \mathcal{A}$ and the initial state $\boldsymbol{x} \in \Omega$.

Define $K\colon \Omega \to \boldsymbol{R}$ as the running cost and $q\colon \partial\Omega \to \boldsymbol{R}$ to be the terminal cost when the state reaches a point on the boundary.[2] Thus, the total cost associated with an initial state $\boldsymbol{x} \in \Omega$ and an admissible control $\boldsymbol{a}(\cdot) \in \mathcal{A}$ is

$$\mathcal{J}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = \int_0^T K(\boldsymbol{y}(s), \boldsymbol{a}(s))\, ds + q(\boldsymbol{y}(T)), \tag{3.3}$$

where $T = \min\{t \mid \boldsymbol{y}(t) \in \partial\Omega\}$. Then the *value function* $u\colon \Omega \to \boldsymbol{R}$ is the minimal total cost to the boundary $\partial\Omega$ starting at $\boldsymbol{x}$:

$$u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)\in\mathcal{A}} \mathcal{J}(\boldsymbol{x}, \boldsymbol{a}(\cdot)). \tag{3.4}$$

By Bellman's optimality principle [7], for every sufficiently small $\tau > 0$,

$$u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)\in\mathcal{A}} \left\{ \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))\, ds + u(\boldsymbol{y}(\tau)) \right\}. \tag{3.5}$$

A Taylor series expansion about $\boldsymbol{x}$ yields the *Hamilton-Jacobi-Bellman equation*:

$$\min_{\boldsymbol{a}\in A} \{K(\boldsymbol{x}, \boldsymbol{a}) + \nabla u(\boldsymbol{x}) \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a})\} = 0. \tag{3.6}$$

Note that, since the minimization is taken over a compact set A, the infimum becomes a minimum. From the definition, the boundary condition on the value function is

$$u(\boldsymbol{x}) = q(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\Omega. \tag{3.7}$$

For the functions $\boldsymbol{f}, K, q$ introduced above, we make the following assumptions:

(A1) $\boldsymbol{f}$ and K are Lipschitz continuous.

(A2) There exist constants K_1, K_2, such that $0 < K_1 \le K(\boldsymbol{x}, \boldsymbol{a}) \le K_2$ for all $\boldsymbol{x} \in \overline{\Omega}$ and $\boldsymbol{a} \in A$.

(A3) The velocity $\boldsymbol{f} : \overline{\Omega} \times A \mapsto \boldsymbol{R}^n$ is bounded (i.e., $\|\boldsymbol{f}\| \le F_2$) and the motion in every direction is always possible; i.e.,

$$\begin{aligned} &\forall \boldsymbol{x} \in \overline{\Omega}, \text{ and all unit vectors } \boldsymbol{v} \in \boldsymbol{R}^n, \\ &\exists \boldsymbol{a} \in A \text{ s.t. } \boldsymbol{v} \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) = \|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a})\| \ge F_1 > 0. \end{aligned}$$

Moreover, we assume that the *scaled vectogram* $\{\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a})/K(\boldsymbol{x}, \boldsymbol{a}) \mid \boldsymbol{a} \in A\}$ is strictly convex at each $\boldsymbol{x} \in \Omega$.

[2] In many applications, the objective is to optimally steer towards a prescribed closed target set $\mathcal{T} \subset \Omega$. This fits into our formulation by setting the domain to be $\Omega \backslash \mathcal{T}$ with the terminal cost $q = \infty$ on $\partial\Omega$ and q finite on $\partial\mathcal{T}$.

(A4) q is lower semicontinuous and $\min_{\partial\Omega} q < \infty$.

The assumption (A3) yields the *small-time controllability* property [3] which guarantees the continuity of the value function u on Ω. In general, even under the aforementioned assumptions, classical solutions to the PDE (3.6) usually do not exist, while weak (Lipschitz-continuous) solutions are not unique. Additional selection criteria were introduced by Crandall and Lions [15] to recover the unique weak solution (the so called *viscosity solution*), coinciding with the value function of the control problem.

Several important classes of examples will repeatedly arise in the following sections. We will refer to problems as having *geometric dynamics* if

$$A = \{\boldsymbol{a} \mid \|\boldsymbol{a}\| = 1\}, \qquad \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) = \boldsymbol{a} f(\boldsymbol{x}, \boldsymbol{a}). \tag{3.8}$$

For control problems with geometric dynamics and unit running cost ($K(\boldsymbol{x}, \boldsymbol{a}) = 1$), the PDE (3.6) reduces to

$$\max_{\|\boldsymbol{a}\|=1} \{f(\boldsymbol{x}, \boldsymbol{a}) \ (-\boldsymbol{a} \cdot \nabla u(\boldsymbol{x}))\} = 1. \tag{3.9}$$

This models a particle which travels through the point $\boldsymbol{x}$ in the direction $\boldsymbol{a} \in A$ at the speed $f(\boldsymbol{x}, \boldsymbol{a})$. For zero terminal cost $q = 0$, the value $u(\boldsymbol{x})$ equals the minimum travel time of such a particle from $\boldsymbol{x}$ to $\partial\Omega$. If we further assume isotropic speed $f(\boldsymbol{x}, \boldsymbol{a}) = f(\boldsymbol{x})$, (3.9) simplifies to the *Eikonal equation*:

$$f(\boldsymbol{x}) \|\nabla u(\boldsymbol{x})\| = 1. \tag{3.10}$$

Finally, for the unit speed with $q = 0$, the viscosity solution to (3.10) is the distance function to $\partial\Omega$.

Under the assumptions (A1)–(A4), the value function u can be used to extract the optimal feedback control $\boldsymbol{a}^* = \boldsymbol{a}^*(\boldsymbol{x})$, provided u is smooth at $\boldsymbol{x} \in \Omega$. For the Eikonal case (3.10), for instance, it can be shown that $\boldsymbol{a}^*(\boldsymbol{x}) = -\nabla u(\boldsymbol{x})/|\nabla u(\boldsymbol{x})|$ (note that $\nabla u \neq 0$ under the assumptions (A2)); the points where ∇u does not exist are precisely those where the optimal control is not unique. Subsequently, the optimal trajectory $\boldsymbol{y}^*(\cdot)$ from $\boldsymbol{x} \in \Omega$ can be computed as the solution to the initial value problem (3.2) with $\boldsymbol{a}(t) = \boldsymbol{a}^*(\boldsymbol{y}(t))$. It is easy to show that $\boldsymbol{y}^*(\cdot)$ coincides with the characteristic curve to $\boldsymbol{x}$ from the boundary $\partial\Omega$, but traveling in the opposite direction. Furthermore, the optimal control will be unique at $\boldsymbol{y}^*(t)$ for all $t > 0$ until this trajectory reaches $\partial\Omega$.

3.2. Augmented PDE for the budget constrained problem. The problem of budget constrained optimal paths was proposed in [27], where general multicriterion optimal control problems were reduced to solving an *augmented PDE* on an expanded state space. For the purpose of this paper, we shall briefly describe this method for the single budget constraint.

Define the extended space $\Omega_e = \Omega \times (0, B]$, where $B > 0$ is a prescribed maximum resource budget. We shall call a point in $(\boldsymbol{x}, b) \in \Omega_e$ an *extended state*. Here, b represents the current resource budget. We represent a path parametrized by time $t \geq 0$ in the extended domain as $\boldsymbol{z}(t) = (\boldsymbol{y}(t), \beta(t)) \in \Omega_e$.

Next, define the secondary cost $\hat{K} \colon \Omega \times A \to \boldsymbol{R}$, strictly positive, which is the decrease rate of the budget. We shall assume that $\hat{K}$ is Lipschitz continuous and there exist constants $\hat{K}_1, \hat{K}_2$ such that

$$0 < \hat{K}_1 \leq \hat{K}(\boldsymbol{x}, \boldsymbol{a}) \leq \hat{K}_2, \quad \forall\, \boldsymbol{x} \in \Omega, \boldsymbol{a} \in A. \tag{3.11}$$

Then the equations of motion in Ω_e are

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \boldsymbol{f}(\boldsymbol{y}(t), \boldsymbol{a}(t)), \qquad \boldsymbol{y}(0) = \boldsymbol{x}, \\ \dot{\beta}(t) &= -\hat{K}(\boldsymbol{y}(t), \boldsymbol{a}(t)), \quad \beta(0) = b. \end{aligned} \tag{3.12}$$

For an arbitrary control $\boldsymbol{a} \in \mathcal{A}$, define the terminal time starting at the extended state $(\boldsymbol{x}, b)$ as

$$\hat{T}(\boldsymbol{x}, b, \boldsymbol{a}) = \min\{t \mid \boldsymbol{y}(t) \in \partial\Omega,\, \beta(t) \geq 0\}.$$

Define the set of all *feasible* controls for paths starting at the extended state $(\boldsymbol{x}, b)$ as

$$\hat{\mathcal{A}}(\boldsymbol{x}, b) = \{\boldsymbol{a} \in \mathcal{A} \mid \hat{T}(\boldsymbol{x}, b, \boldsymbol{a}) < \infty\}.$$

We shall call paths corresponding to feasible controls as feasible paths. Then the value function is

$$w(\boldsymbol{x}, b) = \inf_{\boldsymbol{a}(\cdot) \in \hat{\mathcal{A}}(\boldsymbol{x}, b)} \mathcal{J}(\boldsymbol{x}, \boldsymbol{a}(\cdot)).$$

The boundary condition is

$$w(\boldsymbol{x}, b) = \begin{cases} q(\boldsymbol{x}), & \boldsymbol{x} \in \partial\Omega, b \in [0, B], \\ \infty, & \boldsymbol{x} \in \Omega,\, b = 0. \end{cases} \tag{3.13}$$

Note that $w(\boldsymbol{x}, b) \geq u(\boldsymbol{x})$, since in the unconstrained case the set of controls is larger. Moreover, if the starting budget b is insufficient to reach the boundary, the set $\hat{\mathcal{A}}(\boldsymbol{x}, b)$ is empty and $w(\boldsymbol{x}, b) = \infty$. Wherever the value function is finite on Ω_e, Bellman's optimality principle similarly yields the HJB equation

$$\min_{\boldsymbol{a} \in A} \left\{ \nabla_{\boldsymbol{x}} w \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) - \hat{K}(\boldsymbol{x}, \boldsymbol{a}) \frac{\partial w}{\partial b} + K(\boldsymbol{x}, \boldsymbol{a}) \right\} = 0. \tag{3.14}$$

Here $\nabla_{\boldsymbol{x}}$ denotes the vector-valued operator of partial derivatives with respect to $\boldsymbol{x}$. For the case of isotropic velocity and unit running cost $K = 1$, and $\hat{K}(\boldsymbol{x}, \boldsymbol{a}) = \hat{K}(\boldsymbol{x})$, the latter equation reduces to

$$f(\boldsymbol{x}) |\nabla_{\boldsymbol{x}} w| + \hat{K}(\boldsymbol{x}) \frac{\partial w}{\partial b} = 1. \tag{3.15}$$

The (augmented) velocity function $(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}), -\hat{K}(\boldsymbol{x}, \boldsymbol{a}))$ in (3.12) no longer satisfies the assumption (A3). This implies the lack of small time controllability in Ω_e and possibly discontinuities in the corresponding value function w. While the original definition of viscosity solutions required continuity, a less restrictive notion of *discontinuous viscosity solutions* applies in this case; see [4, Chapter 5], [36], [30] and the references therein.

The assumption (3.11) on $\hat{K}$ implies that the characteristics move in strictly monotonically increasing b-direction (see also Property 3.2). Thus, the value function w is *explicitly causal* in b: the value function at a fixed point $\boldsymbol{x}' \in \Omega$, $b' \in (0, B]$ depends only on the value function in $\{(\boldsymbol{x}, b) \mid b < b'\}$. Moreover, since $\hat{K} > 0$, by rearranging the PDE (3.15)

$$\frac{\partial w}{\partial b} = \frac{1}{\hat{K}(\boldsymbol{x})} \left(1 - f(\boldsymbol{x}) |\nabla_{\boldsymbol{x}} w|\right),$$

the problem can be viewed as a "time-dependent" Hamilton–Jacobi equation, where b represents the "time." In [27], this explicit causality property of (3.14) yielded an efficient numerical scheme involving a single "sweep" in the increasing b-direction (see section 4.3). We note the parallelism with the discrete problem formulation, described in case 2 of Remark 1.

Remark 3. Before moving on to the budget reset problem, we briefly discuss a slight generalization to the budget constrained problem in [27]. Suppose we relax the lower bound in (3.11) to allow $\hat{K} = 0$ on some closed subset $\mathcal{S} \subset \overline{\Omega}$. The set $\mathcal{S}$ can be interpreted as a "safe" set, on which the resources are not used. (In this setting, the problem previously considered in [27] corresponds to $\mathcal{U} = \Omega$.)

Clearly, explicit causality no longer holds when $\mathcal{S} \cap \Omega \neq \emptyset$; rather, w has a *semi-implicit causality property*: for any fixed point $\boldsymbol{x}' \in \Omega$, $b' \in (0, B]$, the value function $w(\boldsymbol{x}', b')$ can only depend on the values $\{w(\boldsymbol{x}, b) \mid \boldsymbol{x} \in \overline{\Omega},\, b \in (0, b']\}$. This possible interdependence between values on the same b level makes it impossible to transform the PDE into a time-dependent Hamilton–Jacobi equation (as with the explicit causality case). Instead, the value function satisfies a separate Eikonal equation on the $\mathrm{int}(\mathcal{S})$ part of each b-slice:

$$\min_{\boldsymbol{a} \in A} \{\nabla_{\boldsymbol{x}} w(\boldsymbol{x}, b) \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) + K(\boldsymbol{x}, \boldsymbol{a})\} = 0, \qquad \forall b \in [0, B], \boldsymbol{x} \in \mathrm{int}(\mathcal{S}). \tag{3.16}$$

We again note the parallelism with the discrete formulation, described in case 3 of Remark 1; see also Figures 2 and 3.

3.3. A PDE formulation for the budget reset problem. We now consider a continuous analogue of the problem in section 2.2 and Figure 4.

Partition the closure of the domain $\bar{\Omega} = \mathcal{U} \cup \mathcal{S}$, where $\mathcal{U}$ is open (thus, $\partial\Omega \subset \mathcal{S}$). Here, $\mathcal{U}$ represents the "unsafe" set, where the budget decreases at some Lipschitz continuous rate $\hat{K}$,

$$0 < \hat{K}_1 \leq \hat{K}(\boldsymbol{x}, \boldsymbol{a}) \leq \hat{K}_2, \qquad \forall\, \boldsymbol{x} \in \mathcal{U}, \boldsymbol{a} \in A, \tag{3.17}$$

and $\mathcal{S}$ represents the "safe" set, where the budget resets to its maximum. We denote the interface between the two sets in the interior of the domain as $\Gamma = \partial\mathcal{U} \cap \Omega$. We will further assume that the set $\mathcal{S} \setminus \{\boldsymbol{x} \in \partial\Omega \mid q(\boldsymbol{x}) < \infty\}$ consists of finitely many connected components.

To model the budget reset property, the budget $\beta(t)$ is set to B in $\mathcal{S}$. Thus, equations (3.12) are modified accordingly in $\mathcal{S}$, and the resulting dynamics is described by a *hybrid system*:

$$\begin{cases} \dot{\boldsymbol{y}}(t) &= \boldsymbol{f}(\boldsymbol{y}(t), \boldsymbol{a}(t)), \\ \dot{\beta}(t) &= -\hat{K}(\boldsymbol{y}(t), \boldsymbol{a}(t)) \end{cases} \qquad \text{if } \boldsymbol{y}(t) \in \mathcal{U}, \tag{3.18}$$

$$\begin{cases} \dot{\boldsymbol{y}}(t) &= \boldsymbol{f}(\boldsymbol{y}(t), \boldsymbol{a}(t)), \\ \beta(t) &= B \end{cases} \qquad \text{if } \boldsymbol{y}(t) \in \mathcal{S}, \tag{3.19}$$

with initial states

$$\beta(0) = \begin{cases} b & \text{if } \boldsymbol{y}(0) \in \mathcal{U}, \\ B & \text{if } \boldsymbol{y}(0) \in \mathcal{S}. \end{cases} \tag{3.20}$$

Similarly to section 3.2, we define the terminal time for a control $\boldsymbol{a} \in \mathcal{A}$ as

$$\tilde{T}(\boldsymbol{x}, b, \boldsymbol{a}) = \min\{t \mid \boldsymbol{y}(t) \in \partial\Omega, \beta(\tau) \geq 0 \quad \forall\, \tau \in [0, t]\},$$

where the paths $\boldsymbol{y}(\cdot)$ satisfy (3.18)–(3.20).

Let $\tilde{\mathcal{A}}(\boldsymbol{x}, b) = \{\boldsymbol{a} \in \mathcal{A} \mid \tilde{T}(\boldsymbol{x}, b, \boldsymbol{a}) < \infty\}$ be the set of all feasible controls for the budget reset problem. Then the value function is defined as

$$w(\boldsymbol{x}, b) = \inf_{\boldsymbol{a}(\cdot) \in \tilde{\mathcal{A}}(\boldsymbol{x}, b)} \mathcal{J}(\boldsymbol{x}, \boldsymbol{a}(\cdot)).$$

For the budget reset problem, a path $\boldsymbol{z}(\cdot) = (\boldsymbol{y}(\cdot), \beta(\cdot))$ in the extended domain never visits points $\{(\boldsymbol{x}, b) \mid \boldsymbol{x} \in \mathcal{S}, b < B\}$. Thus, we consider the *reduced extended domain*

$$\Omega_r = \Omega_{\mathcal{U}} \cup \Omega_{\mathcal{S}} \ \subset \Omega_e,$$

where

$$\Omega_{\mathcal{U}} = \mathcal{U} \times (0, B],$$
$$\Omega_{\mathcal{S}} = \mathcal{S} \times \{B\}.$$

This reduction of the extended state space is analogous to the discrete example described in Figure 4. The corresponding HJB equation for this *hybrid control problem* in Ω_r can be described separately in $\Omega_{\mathcal{U}}$ and $\Omega_{\mathcal{S}}$. To distinguish between the value functions in these two domains we write

$$w(\boldsymbol{x}, b) \equiv \begin{cases} w_1(\boldsymbol{x}, b), & (\boldsymbol{x}, b) \in \Omega_{\mathcal{U}}, \\ w_2(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{S}. \end{cases}$$

As in the no-resets case, the value function w is usually infinite on parts of Ω_r (if the starting budget is insufficient to reach the boundary even with resets); see also section 4.4. Bellman's optimality principle can be used to show that, wherever w is finite, it should satisfy the following PDEs:

$$\min_{\boldsymbol{a} \in A} \left\{ \nabla w_1(\boldsymbol{x}, b) \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) - \hat{K}(\boldsymbol{x}, \boldsymbol{a}) \frac{\partial w_1}{\partial b}(\boldsymbol{x}, b) + K(\boldsymbol{x}, \boldsymbol{a}) \right\} = 0, \quad (\boldsymbol{x}, b) \in \Omega_{\mathcal{U}}, \tag{3.21}$$

$$\min_{\boldsymbol{a} \in A} \{\nabla w_2(\boldsymbol{x}) \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) + K(\boldsymbol{x}, \boldsymbol{a})\} = 0, \quad \boldsymbol{x} \in \operatorname{int}(\mathcal{S}). \tag{3.22}$$

We note the similarity between the previously defined PDE (3.16) and (3.22). But while the latter is solved on one b-slice only (for $b = B$), the former is actually a b-parametrized family of static PDEs, which are coupled to each other only through the boundary with $\Omega_{\mathcal{U}}$.

This difference is a direct consequence of the budget reset property.

The boundary conditions on w are

$$w(\boldsymbol{x}, b) = \begin{cases} q(\boldsymbol{x}), & \boldsymbol{x} \in \partial\Omega \subset \mathcal{S}, \\ \infty, & \boldsymbol{x} \in \Omega, b = 0. \end{cases} \tag{3.23}$$

In addition, the following compatibility condition (motivated below by Proposition 3.3) holds between w_1 and w_2, wherever w is finite on $\Gamma = \partial\mathcal{U} \cap \Omega$:

$$\lim_{\substack{\boldsymbol{x}' \to \boldsymbol{x} \\ \boldsymbol{x}' \in \mathcal{U}}} w_1(\boldsymbol{x}', b) = w_2(\boldsymbol{x}), \qquad \forall\, \boldsymbol{x} \in \Gamma, b \in (0, B]. \tag{3.24}$$

We now list several properties of the value function w. The proofs of the first two of them are omitted for brevity.

PROPERTY 3.1. *In $\Omega_{\mathcal{U}}$, the b-coordinate is monotonically decreasing along every trajectory when traversing forward in time.*

PROPERTY 3.2. *If $b_1 \le b_2 \le B$, then $w_1(\boldsymbol{x}, b_1) \ge w_1(\boldsymbol{x}, b_2)$ for all $\boldsymbol{x} \in \mathcal{U}$.*

From here on we will use $\mathcal{S}_{\mathcal{R}} = \{\boldsymbol{x} \in \mathcal{S} \mid w_2(\boldsymbol{x}) < \infty\}$ to denote the "reachable" part of the safe set. We note that every connected component of $\mathcal{S} \setminus \{\boldsymbol{x} \in \partial\Omega \mid q(\boldsymbol{x}) < \infty\}$ is either fully in $\mathcal{S}_{\mathcal{R}}$ or does not intersect it at all. As a result, the number of connected components in the latter set is also finite.

LEMMA 3.1. *w_2 is locally Lipschitz continuous on $\mathcal{S}_{\mathcal{R}} \backslash \partial\Omega$.*

Proof. We will use the notation $B_\epsilon(\boldsymbol{x}) = \{\boldsymbol{x}' \in \boldsymbol{R}^n \mid \|\boldsymbol{x} - \boldsymbol{x}'\| < \epsilon\}$. Choose a point $\boldsymbol{x} \in \mathcal{S}_{\mathcal{R}} \backslash \partial\Omega$, and an $\epsilon > 0$ sufficiently small so that $B_\epsilon(\boldsymbol{x}) \subset \Omega$ and

$$\frac{\epsilon}{F_1} < \frac{B}{\hat{K}_2}. \tag{3.25}$$

Take any $\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x}) \cap (\mathcal{S}_{\mathcal{R}} \backslash \partial\Omega)$. Then, a straight line path from $\boldsymbol{x}$ to $\boldsymbol{x}'$ will take at most $\|\boldsymbol{x} - \boldsymbol{x}'\|/F_1$ time to traverse. Furthermore, by (3.25), such a path must be feasible. Thus, Bellman's optimality principle gives $w_2(\boldsymbol{x}') \le K_2/F_1 \|\boldsymbol{x} - \boldsymbol{x}'\| + w_2(\boldsymbol{x})$. By swapping the roles of $\boldsymbol{x}$ and $\boldsymbol{x}'$ we have $|w_2(\boldsymbol{x}') - w_2(\boldsymbol{x})| \le K_2/F_1 \|\boldsymbol{x} - \boldsymbol{x}'\|$, as desired. ∎

A consequence of Lemma 3.1 is that w_2 is bounded on each connected component of $\mathcal{S}_{\mathcal{R}}$, excluding $\partial\Omega$. Since $\mathcal{S}_{\mathcal{R}}$ consists of finitely many connected components, $w_2^{\max} = \sup_{\boldsymbol{x} \in \mathcal{S}_{\mathcal{R}} \backslash \partial\Omega} w_2(\boldsymbol{x})$ is also finite.

LEMMA 3.2. *For a point $\boldsymbol{x} \in \mathcal{U}$ such that $w_1(\boldsymbol{x}, B) < \infty$, let $\boldsymbol{y}^*$ be an optimal feasible path from $\boldsymbol{x}$ to $\partial\Omega$. Let $T(\boldsymbol{x})$ be the first arrival time of $\boldsymbol{y}^*$ to $\partial\Omega$. Then,*

$$T(\boldsymbol{x}) \le \frac{B}{\hat{K}_1} + \frac{w_2^{\max}}{K_1}. \tag{3.26}$$

Proof. Let $t^* > 0$ be the first instance in time such that $\boldsymbol{y}^*(t^*) \in \mathcal{S}$. We subdivide $\boldsymbol{y}^*$ into two segments, the first from $\boldsymbol{x}$ to $\boldsymbol{y}^*(t^*)$ and the second from $\boldsymbol{y}^*(t^*)$ to the final point $\boldsymbol{y}^*(T(\boldsymbol{x})) \in \partial\Omega$. The time taken to traverse the first segment is bounded by $B/\hat{K}_1$ to satisfy the budget feasibility constraint, and the time on the second segment is bounded by $w_2^{\max}/K_1$. ∎

PROPOSITION 3.3. *The compatibility condition* (3.24) *holds on $\partial\mathcal{S}_{\mathcal{R}} \backslash \partial\Omega$ for $b \in (0, B)$.*

Proof. Fix a point $\boldsymbol{x} \in \partial\mathcal{S}_{\mathcal{R}} \backslash \partial\Omega$. Choose an $\epsilon_1 > 0$ sufficiently small so that $B_{\epsilon_1}(\boldsymbol{x}) \subset \Omega$ and $\epsilon_1 < bF_1/\hat{K}_2$. If we choose $\boldsymbol{x}' \in \mathcal{U}$ such that $\|\boldsymbol{x}' - \boldsymbol{x}\| < \epsilon_1$, by arguments similar to the proof of Lemma 3.1, the straight line path from $\boldsymbol{x}'$ to $\boldsymbol{x}$ is feasible. Thus, by Bellman's optimality principle,

$$w_1(\boldsymbol{x}', b) \le w_2(\boldsymbol{x}) + \epsilon_1 K_2/F_1 \quad \text{for } b \in (0, B]. \tag{3.27}$$

Suppose now that $b \in (0, B)$. Choose $\epsilon_2 > 0$ so that $B_{\epsilon_2}(\boldsymbol{x}) \subset \Omega$ and $\epsilon_2 < \frac{(B-b)F_1}{\hat{K}_2}$. Assume $\boldsymbol{x}' \in \mathcal{U}$ and $\|\boldsymbol{x} - \boldsymbol{x}'\| < \epsilon_2$, and consider the straight line path from $\boldsymbol{x}$ to $\boldsymbol{x}'$. Suppose the resource cost and time required to traverse this path is b' and τ, respectively. Since $\tau \le \epsilon_2/F_1$, our choice of ϵ_2 ensures that $b' \le \tau\hat{K}_2 \le (\epsilon_2/F_1)\,\hat{K}_2 < B - b$, or equivalently, $b < B - b'$.

Thus, by Bellman's optimality principle,

$$\begin{aligned} w_2(\boldsymbol{x}) &\le w_1(\boldsymbol{x}', B - b') + \epsilon_2 K_2/F_1 \\ &\le w_1(\boldsymbol{x}', b) + \epsilon_2 K_2/F_1 \quad \text{for } b \in (0, B), \end{aligned} \tag{3.28}$$

where the second inequality follows from Proposition 3.2. Therefore, (3.27) and (3.28) imply that if $\|\boldsymbol{x}' - \boldsymbol{x}\| < \epsilon = \min\{\epsilon_1, \epsilon_2\}$, then $|w_1(\boldsymbol{x}', b) - w_2(\boldsymbol{x})| \le \epsilon K_2/F_1$ for $\boldsymbol{x}' \in \mathcal{U}, b \in (0, B)$; this proves the compatibility condition (3.24) for $b \in (0, B)$. ∎

PROPOSITION 3.4. *Assume the running costs and the dynamics are isotropic; i.e.,* $K(\boldsymbol{x}, \boldsymbol{a}) = K(\boldsymbol{x})$, $\hat{K}(\boldsymbol{x}, \boldsymbol{a}) = \hat{K}(\boldsymbol{x})$, *and* $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) = \boldsymbol{a} f(\boldsymbol{x}), \|\boldsymbol{a}\| = 1$. *Then the compatibility condition* (3.24) *holds on* $\partial\mathcal{S}_{\mathcal{R}} \backslash \partial\Omega$ *for* $b = B$.

Proof. Fix $\boldsymbol{x} \in \partial\mathcal{S}_{\mathcal{R}} \backslash \partial\Omega$. We prove that $w_2(\boldsymbol{x}) - \lim_{\boldsymbol{x}' \to \boldsymbol{x}} w_1(\boldsymbol{x}', B) \le 0$. Note that (3.24) follows, since the proof of (3.27) also covered the current case ($b = B$).

Take a sequence $\{\boldsymbol{x}_j\} \subset \mathcal{U}$ converging to $\boldsymbol{x}$, such that $w_1(\boldsymbol{x}_j, B)$ is finite for each j. Suppose for each j that $\boldsymbol{a}_j^* \in \tilde{\mathcal{A}}(\boldsymbol{x}_j, B)$ is an optimal feasible control and $\boldsymbol{y}_j^*$ is the corresponding feasible path such that $T_j = \min\{t \mid \boldsymbol{y}_j^*(t) \in \partial\Omega\}$. Let $T_{\max} = \sup_j\{T_j\}$ and for each j, set $\boldsymbol{y}_j^*(t) = \boldsymbol{y}_j^*(T_j)$ for $t \in [T_j, T_{\max}]$. Note that by Lemma 3.2, $T_{\max}$ is finite. Also, condition (A3) yields uniform boundedness and equi-continuity of the paths $\boldsymbol{y}_j^*$. Therefore, the Arzela–Ascoli theorem ensures that (upon reindexing in j) a subsequence $\boldsymbol{y}_j^*$ converges uniformly to a path $\boldsymbol{y}^*$ in $\overline{\Omega}$ corresponding to some admissible control $\boldsymbol{a}^* \in \mathcal{A}$.

Next, we show that $\boldsymbol{a}^* \in \tilde{\mathcal{A}}(\boldsymbol{x}', B)$; i.e., $\boldsymbol{y}^*$ is a feasible path. Define T_j^* (and T^*) to be the first instance in time when $\boldsymbol{y}_j^*$ (respectively, $\boldsymbol{y}^*$) reaches $\partial\mathcal{S}_{\mathcal{R}}$. Let $\tilde{T}$ be the infimum limit of the sequence $\{T_j^*\}$ and consider its subsequence such that, upon reindexing in j, $\tilde{T} = \lim_{j\to\infty} T_j^*$. Since $\partial\mathcal{S}_{\mathcal{R}}$ is closed, $\boldsymbol{y}^*(\tilde{T}) = \lim_{j\to\infty} \boldsymbol{y}_j^*(T_j^*) \in \partial\mathcal{S}_{\mathcal{R}}$, and thus $\tilde{T} \ge T^*$. This implies that $\lim_{j\to\infty} \int_{T^*}^{T_j^*} \hat{K}(\boldsymbol{y}_j^*(s))ds \ge 0$. Using this observation and the Lipschitz continuity of $\hat{K}$, we have

$$\begin{aligned}
\int_0^{T^*} \hat{K}(\boldsymbol{y}^*(s))ds - \int_0^{T_j^*} \hat{K}(\boldsymbol{y}_j^*(s))ds &= \int_0^{T^*} \hat{K}(\boldsymbol{y}^*(s)) - \hat{K}(\boldsymbol{y}_j^*(s))ds - \int_{T^*}^{T_j^*} \hat{K}(\boldsymbol{y}_j^*(s))ds \\
&\le \int_0^{T^*} \left|\hat{K}(\boldsymbol{y}^*(s)) - \hat{K}(\boldsymbol{y}_j^*(s))\right| ds - \int_{T^*}^{T_j^*} \hat{K}(\boldsymbol{y}_j^*(s))ds \\
&\le L \int_0^{T^*} \left\|\boldsymbol{y}^*(s) - \boldsymbol{y}_j^*(s)\right\| ds - \int_{T^*}^{T_j^*} \hat{K}(\boldsymbol{y}_j^*(s))ds,
\end{aligned}$$

where L is the Lipschitz constant for $\hat{K}(\cdot)$. Since the last expression has a nonpositive limit as $j \to \infty$ and each $\boldsymbol{y}_j^*$ is feasible, this shows that on the interval $[0, T^*]$ the trajectory $\boldsymbol{y}^*$ is feasible as well:

$$\int_0^{T^*} \hat{K}(\boldsymbol{y}^*(s))ds \le \lim_{j\to\infty} \int_0^{T_j^*} \hat{K}(\boldsymbol{y}_j^*(s))ds \le B.$$

If $\boldsymbol{y}^*(t)$ does not remain in $\mathcal{S}_{\mathcal{R}}$ after $t = T^*$, a similar argument can be used to prove the feasibility of all other "unsafe segments" of the trajectory. (Alternatively, appending the optimal trajectory corresponding to $\boldsymbol{y}^*(T^*) \in \mathcal{S}_{\mathcal{R}}$ yields another feasible trajectory, which is at least as cheap with regard to the primary running cost K.)

Applying the same argument to the total running cost (3.3) and recalling that q is lower semicontinuous, we obtain $\mathcal{J}(\boldsymbol{x}, \boldsymbol{a}^*) \le \lim_{j\to\infty} w_1(\boldsymbol{x}_j, B)$. This completes the proof since, by definition of the value function, $w_2(\boldsymbol{x}) \le \mathcal{J}(\boldsymbol{x}, \boldsymbol{a}^*)$. ∎

4. Numerical methods for continuous budget reset problems. Throughout this section, we assume the following setup: let $\mathcal{G}$ be a set of gridpoints on the domain $\bar{\Omega}$. While a similar formulation can be constructed for arbitrary unstructured meshes, we restrict our description to a uniform Cartesian grid $\mathcal{G}$ with grid spacing h.

We denote $\mathcal{G}_{\mathcal{U}} = \mathcal{G} \cap \mathcal{U}$ and $\mathcal{G}_{\mathcal{S}} = \mathcal{G} \cap \mathcal{S}$. To simplify the treatment of boundary conditions, we assume that $\partial\Omega$ and Γ are well-discretized by the gridpoints in $\partial\mathcal{G} \subset \mathcal{G}$ and in $\partial\mathcal{G}_{\mathcal{S}}$, respectively. We also assume that the set of allowable budgets $[0, B]$ is discretized into equispaced intervals partitioned by gridpoints $\mathcal{B} = \{b_j = j\Delta b \mid j = 0, 1, \ldots, N_b\}$, where $\Delta b > 0$ is fixed ahead of time.

4.1. Discretization of the unconstrained case. To begin, we briefly discuss the usual semi-Lagrangian discretization techniques for static HJB equations of the form (3.6). Denote $U(\boldsymbol{x})$ to be the numerical approximation to $u(\boldsymbol{x})$ at the gridpoint $\boldsymbol{x} \in \mathcal{G}$. Suppose the current system state is $\boldsymbol{x} \in \mathcal{G} \cap \Omega$ and the constant control value $\boldsymbol{a}$ is to be used for a short time $\tau > 0$. Assuming that K and $\boldsymbol{f}$ are locally constant, the new position is approximated by $\boldsymbol{x}_{\boldsymbol{a}}(\tau) = \boldsymbol{x} + \tau \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a})$ and the approximate accumulated transition cost is $K(\boldsymbol{x}, \boldsymbol{a})\tau$. For small τ, this yields a first-order *semi-Lagrangian* discretization

$$
\begin{aligned}
U(\boldsymbol{x}) &= \min_{\boldsymbol{a} \in A} \{\tau K(\boldsymbol{x}, \boldsymbol{a}) + U(\boldsymbol{x}_{\boldsymbol{a}}(\tau))\}, \qquad \forall\, \boldsymbol{x} \in \mathcal{G} \backslash \partial\mathcal{G}, \\
U(\boldsymbol{x}) &= q(\boldsymbol{x}), \qquad\qquad\qquad\qquad\qquad\quad \forall\, \boldsymbol{x} \in \partial\mathcal{G}
\end{aligned} \tag{4.1}
$$

of Bellman's optimality principle (3.5). Since $\boldsymbol{x}_{\boldsymbol{a}}(\tau)$ is usually not a gridpoint, $U(\boldsymbol{x}_{\boldsymbol{a}}(\tau))$ needs to be interpolated using adjacent grid values.

Many different variants of the above scheme result from different choices of τ. Falcone and coauthors have extensively studied the discretized systems[3] which use the same $\tau > 0$ for all $\boldsymbol{x}$ and $\boldsymbol{a}$; see [3, 19, 20] and higher-order accurate versions in [21]. Alternatively, τ can be chosen for each $\boldsymbol{a}$ to ensure that $\boldsymbol{x}_{\boldsymbol{a}}(\tau)$ lies on some prespecified set near $\boldsymbol{x}$. For example, in a version considered by Gonzales and Rofman [23], the motion continues in the chosen direction until reaching the boundary of an adjacent simplex. For example, on a Cartesian grid in $\boldsymbol{R}^2$, if $\boldsymbol{x}_s$ and $\boldsymbol{x}_w$ are two gridpoints adjacent to $\boldsymbol{x}$ and $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a})$ is some southwest-ward direction of motion, then τ is chosen to ensure that $\boldsymbol{x}_{\boldsymbol{a}}(\tau)$ lies on a segment $\boldsymbol{x}_s\boldsymbol{x}_w$, and $U(\boldsymbol{x}_{\boldsymbol{a}}(\tau))$ is approximated by a linear interpolation between $U(\boldsymbol{x}_s)$ and $U(\boldsymbol{x}_w)$. Interestingly, in the case of geometric dynamics (3.8), this type of semi-Lagrangian scheme is also equivalent to the usual Eulerian (upwind finite difference) discretization. A detailed discussion of this for isotropic problems on grids can be found in [40] and for general anisotropic problems on grids and meshes in [35, Appendix]. In addition, both types of semi-Lagrangian schemes can be viewed as controlled Markov processes on $\mathcal{G}$; this earlier approach was pioneered by Kushner and Dupuis in [28]; see also a more recent discussion in [41] on applicability of label-setting algorithms.

We note that (4.1) is a large coupled system of nonlinear equations. If Ψ is an upper bound on U, this system can, in principle, be solved by fixed point iterations starting from an initial guess $U = \Psi$ on $\mathcal{G} \backslash \partial\mathcal{G}$. However, this approach can be computationally expensive, and an attractive alternative is to develop a Dijkstra-like noniterative algorithm. For the fully isotropic case, two such methods are the Tsitsiklis' algorithm [40] and the Sethian Fast Marching Method [31]. An overview of many (isotropic) extensions of this approach can be found in [33]. Ordered Upwind Methods [34, 35] have similarly handled the anisotropic case; a recent efficient modification was introduced in [2]. Similar fast methods were introduced for several related PDEs by Falcone and collaborators [16, 10, 11, 12].

[3]Some of the papers cited in this subsection have considered related but slightly different PDEs, including those for finite-horizon and infinite-horizon optimal control, but the fundamental idea remains the same.

An alternative approach is to speed up the convergence of iterative solver for (4.1) by using Gauss–Seidel iterations with an alternating ordering of gridpoints. Such "Fast Sweeping" algorithms [9, 39, 42, 26] are particularly efficient when the direction of characteristics does not change too often. A comparison of various noniterative and fast-iterative approaches (as well as a discussion of more recent hybrid algorithms) can be found in [13].

We emphasize that, for the purposes of this paper, any one of the above approaches can be used modularly, whenever we need to solve (3.22). The discretization of (3.21) is explained in subsection 4.3. But before dealing with these technical details, subsection 4.2 addresses the main computational challenge of budget reset problems: the a priori unknown boundary condition on Γ, which results in an implicit interdependence of gridpoints in $\Omega_{\mathcal{U}}$ and $\Omega_{\mathcal{S}}$.

4.2. Iterative treatment of the budget reset problem. First, we note that the simpler case of $\Omega = \mathcal{U}$ (i.e., the constrained optimal control problem presented in section 3.2) can be solved by a single upward sweep in the b-direction, as described in [27]. This is a direct consequence of the explicit causality property of the value function when $\hat{K}$ is strictly positive on Ω. Moreover, without budget resets, relaxing $\hat{K}$ to be nonnegative (i.e., introducing a safe subset $\mathcal{S}$ where $\hat{K} = 0$) still yields semi-implicit causality; see Remark 3.

In contrast, the introduction of budget resets on a safe subset $\mathcal{S} \subset \Omega$ breaks this causal structure. If the values on Γ were a priori known, we could efficiently solve (3.22) on $\mathcal{S}$ by either Marching or Sweeping techniques, at least in the case of geometric dynamics. But since the values on Γ are not provided, there are no known noniterative algorithms to numerically solve this problem on Ω_r. Therefore, we propose solving the PDEs (3.21) and (3.22) (with boundary conditions and the compatibility condition (3.24)) by an alternating iterative process. We construct a sequence of functions w_1^k and w_2^k for $k = 0, 1, 2, \ldots$, which converge to w_1 and w_2, respectively, as $k \to \infty$.

We begin with a recursive definition for these new functions on $\Omega_{\mathcal{U}}$ and $\Omega_{\mathcal{S}}$. Of course, the actual implementation in section 4.4 relies on their numerical approximations; the resulting method is summarized in Algorithm 2. Initially, set

$$\begin{aligned} w_1^0(\boldsymbol{x}, b) &= \infty, \quad (\boldsymbol{x}, b) \in \Omega_{\mathcal{U}}, \\ w_2^0(\boldsymbol{x}) &= \infty, \qquad\quad \boldsymbol{x} \in \mathcal{S}. \end{aligned}$$

Then for $k = 1, 2, \ldots$, we have the following phases:

Phase I. Find w_1^k as the viscosity solution of (3.21) with boundary conditions

$$w_1^k(\boldsymbol{x}, b) = \begin{cases} q(\boldsymbol{x}), & \boldsymbol{x} \in \partial\Omega, b \in (0, B], \\ \liminf\limits_{\substack{\boldsymbol{x}' \to \boldsymbol{x} \\ \boldsymbol{x}' \in \mathcal{S}}} w_2^{k-1}(\boldsymbol{x}'), & \boldsymbol{x} \in \Gamma, b \in (0, B], \\ \infty, & \boldsymbol{x} \in \mathcal{U}, b = 0. \end{cases} \tag{4.2}$$

Phase II. Find w_2^k as the viscosity solution of (3.22) with boundary conditions

$$w_2^k(\boldsymbol{x}) = \begin{cases} q(\boldsymbol{x}), & \boldsymbol{x} \in \partial\Omega, \\ \liminf\limits_{\substack{\boldsymbol{x}' \to \boldsymbol{x} \\ \boldsymbol{x}' \in \mathcal{U}}} w_1^k(\boldsymbol{x}', B), & \boldsymbol{x} \in \Gamma. \end{cases} \tag{4.3}$$

We note that the lim infs in the above definition are primarily for the sake of notational consistency (since solving (3.22) on $\text{int}(\mathcal{S})$ does not really specify w_2^k's values on $\Gamma \subset \partial\mathcal{S}$. Alternatively, we can solve the PDE on $\mathcal{S}$, treating boundary conditions on Γ "in the viscosity sense" [4]. This is essentially the approach used in our numerical implementation.

Intuitively, w_1^k and w_2^k can be interpreted as answering the same question as w_1 and w_2, but with an additional constraint that no trajectory is allowed to reset the budget (by crossing from $\mathcal{U}$ to $\mathcal{S}$) more than $(k-1)$ times. As a result, for many problems convergence is attained (i.e., $w_1^k = w_1$ and $w_2^k = w_2$) after a finite number of recursive steps. For example, in the simplest case where K and f are constant on Ω, $\hat{K} > 0$ is constant on $\mathcal{U}$, and all connected components of $\mathcal{S}\backslash\partial\Omega$ are convex, then any optimal trajectory might enter each connected component at most once. See Table 2 for the experimental confirmation of this phenomenon.

4.3. Discretization of w_2^k, w_1^k. Let $W_1^k(\boldsymbol{x}, b_j)$ be an approximation of $w_1^k(\boldsymbol{x}, b_j)$ for all $(\boldsymbol{x}, b_j) \in \mathcal{G}_\mathcal{U} \times \mathcal{B}$; similarly, let $W_2^k(\boldsymbol{x})$ be an approximation of $w_2^k(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{G}_\mathcal{S}$. The "natural" boundary conditions are implemented as follows:

$$W_1^k(\boldsymbol{x}, b_j) = q(\boldsymbol{x}), \qquad \boldsymbol{x} \in \partial\mathcal{G} \cap \bar{\mathcal{U}}, b_j \in \mathcal{B}, \tag{4.4}$$

$$W_2^k(\boldsymbol{x}) = q(\boldsymbol{x}), \qquad \boldsymbol{x} \in \partial\mathcal{G} \cap \mathcal{S}. \tag{4.5}$$

Additional boundary conditions on Γ stem from the recursive definition of w_2^k and w_1^k (yielding the compatibility condition (3.24) in the limit). In Phase I, we use

$$W_1^k(\boldsymbol{x}, b_j) = W_2^{k-1}(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathcal{G}_\mathcal{S}, b_j \in \mathcal{B}, \tag{4.6}$$

and then solve the discretized system (4.8) on the relevant subset of $\mathcal{G}_\mathcal{U} \times \mathcal{B}$; see the discussion below and in section 4.4.

In Phase II, the numerical compatibility condition is enforced on the set of gridpoints $\mathcal{G}_\mathcal{U}^\Gamma = \{\boldsymbol{x} \in \mathcal{G}_\mathcal{U} \mid \boldsymbol{x} \text{ is adjacent to some } \boldsymbol{x}' \in \mathcal{S}\}$. We set

$$W_2^k(\boldsymbol{x}) = W_1^k(\boldsymbol{x}, B), \quad \boldsymbol{x} \in \mathcal{G}_\mathcal{U}^\Gamma, \tag{4.7}$$

and then recover W_2^k by solving the system of equations equivalent to (4.1) on the entire $\mathcal{G}_\mathcal{S}$ (including on $\mathcal{G} \cap \Gamma$). As explained in subsection 4.1, this can be accomplished by many different efficient numerical algorithms.

To derive the system of equations defining W_1^k on $\mathcal{G}_\mathcal{U} \times \mathcal{B}$, we adapt the approach introduced in [27]. Property 3.1 is fundamental for the method's efficiency: the characteristic curves emanating from $\partial\mathcal{S}_\mathcal{R}$ all move in increasing direction in b. Thus, we need only a single "upward" sweep in the b direction to capture the value function along the characteristics. We exploit this result in the semi-Lagrangian framework as follows. For $\boldsymbol{x} \in \mathcal{G} \cap \mathcal{U}$, $b_j \in \mathcal{B}$, $\tau > 0$, write $\boldsymbol{x}_{\boldsymbol{a}}(\tau) = \boldsymbol{x} + \tau \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a})$ and $b_{\boldsymbol{a}}(\tau) = b_j - \tau\hat{K}(\boldsymbol{x}, \boldsymbol{a})$. If we choose $\tau = \tau_{\boldsymbol{a},\boldsymbol{x}} = (\Delta b)/\hat{K}(\boldsymbol{x}, \boldsymbol{a})$, this ensures that $b_{\boldsymbol{a}}(\tau) = b_{j-1}$, and the semi-Lagrangian scheme at $(\boldsymbol{x}, b_j)$ becomes

$$W_1^k(\boldsymbol{x}, b_j) = \min_{\boldsymbol{a} \in A} \left\{ \tau K(\boldsymbol{x}, \boldsymbol{a}) + W_1^k(\boldsymbol{x}_{\boldsymbol{a}}(\tau), b_{j-1}) \right\}. \tag{4.8}$$

For each $j = 1, 2, \ldots$, we solve (4.8) for $W_1^k(\boldsymbol{x}, b_j)$ at each $\boldsymbol{x} \in \mathcal{G}_\mathcal{U}$ based on the (already computed) W_1^k values in the b_{j-1} resource-level.

For an arbitrary control value $\boldsymbol{a} \in A$, the point $\boldsymbol{x}_{\boldsymbol{a}}(\tau)$ usually is not a gridpoint; so, $W_1^k(\boldsymbol{x}_{\boldsymbol{a}}(\tau), b_{j-1})$ has to be approximated using values at the nearby gridpoints (some

of which may be in $\mathcal{G}_\mathcal{S}$). Since our approximation of $\boldsymbol{x_a}(\tau)$ is first-order accurate, it is also natural to use a first-order approximation for its W_1^k value. Our implementation relies on a bilinear interpolation. For example, for $n = 2$, suppose the gridpoints $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4 \in \mathcal{G}$ are the four corners of the grid cell containing $\boldsymbol{x_a}(\tau)$, ordered counterclockwise with $\boldsymbol{x}_1$ on the bottom left corner. If $(\gamma_1, \gamma_2) = (\boldsymbol{x_a}(\tau) - \boldsymbol{x}_1)/h$, then the bilinear interpolation yields

$$\begin{aligned} W_1^k(\boldsymbol{x_a}(\tau), b_{j-1}) = \gamma_1 \left(\gamma_2 \mathcal{W}^k(\boldsymbol{x}_3, b_{j-1}) + (1-\gamma_2)\mathcal{W}^k(\boldsymbol{x}_2, b_{j-1})\right) \\ + (1-\gamma_1)\left(\gamma_2 \mathcal{W}^k(\boldsymbol{x}_4, b_{j-1}) + (1-\gamma_2)\mathcal{W}^k(\boldsymbol{x}_1, b_{j-1})\right), \end{aligned}$$

where $\mathcal{W}^k(\boldsymbol{x}, b) = W_1^k(\boldsymbol{x}, b)$ if $\boldsymbol{x} \in \mathcal{G}_\mathcal{U}$ and $\mathcal{W}^k(\boldsymbol{x}, b) = W_2^{k-1}(\boldsymbol{x})$ if $\boldsymbol{x} \in \mathcal{G}_\mathcal{S}$. The resulting approximation is inherently continuous, while w_1 usually is not. The issue of convergence of semi-Lagrangian schemes to discontinuous viscosity solutions is discussed in Remark 4.

The direct discretization of (3.21) and (3.22) could also be interpreted as defining the value function of the corresponding *Stochastic Shortest Path* problem on $(\mathcal{G}_\mathcal{U} \times \mathcal{B}) \cup \mathcal{G}_\mathcal{S}$; see section 3 in [37]. In this framework, the iterative algorithm presented in this section can be viewed as a Gauss–Seidel relaxation of value iterations on $\mathcal{G}_\mathcal{U} \times \mathcal{B}$ alternating with a Dijkstra-like method used on $\mathcal{G}_\mathcal{S}$.

4.4. Domain restriction techniques. The value function $w(\boldsymbol{x}, b)$ is infinite at points that are not reachable within budget b. Since the dimension $(n + 1)$ of the domain where w_1 is solved is typically large, a reduction of the computational domain to its reachable subset usually yields substantial saving in computational time. In [27] such a reduction was achieved by an efficient preprocessing step, which involved computing the *minimum feasible level*, i.e. the interface that separates the reachable and unreachable parts of Ω_e. Here we use a similar technique to find the "lower" boundary of the reachable subset of $\Omega_\mathcal{U}$.

Note that, by Property 3.2, for any $\boldsymbol{x} \in \mathcal{U}$, $w_1(\boldsymbol{x}, b_1) < \infty$ implies $w_1(\boldsymbol{x}, b) < \infty$ for all $b \in [b_1, B]$; and $w_1(\boldsymbol{x}, b_2) = \infty$ implies $w_1(\boldsymbol{x}, b) = \infty$ for all $b \in [0, b_2]$. We formally define the minimum feasible level (MFL) in the unsafe set as the graph of

$$v(\boldsymbol{x}) = v[w_1](\boldsymbol{x}) = \min\{b \mid w_1(\boldsymbol{x}, b) < \infty\}, \quad \boldsymbol{x} \in \mathcal{U}, \tag{4.9}$$

and in the safe set $\mathcal{S}$, as a graph of

$$v(\boldsymbol{x}) = \begin{cases} 0, & \boldsymbol{x} \in \mathcal{S}_\mathcal{R}, \\ \infty, & \boldsymbol{x} \in \mathcal{S} \backslash \mathcal{S}_\mathcal{R}. \end{cases} \tag{4.10}$$

The goal is to recover the MFL from some cheaper (lower-dimensional) computation. We note that on $\mathcal{U}$, $v(\boldsymbol{x})$ can be interpreted as the value function of another resource-optimal control problem, and as such it can be recovered as the viscosity solution of a similar HJB equation

$$\min_{\boldsymbol{a} \in A} \left\{ \hat{K}(\boldsymbol{x}, \boldsymbol{a}) + \nabla v(\boldsymbol{x}) \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) \right\} = 0, \quad \boldsymbol{x} \in \mathcal{U}, \tag{4.11}$$

coupled with the boundary conditions (4.10). We note that $\mathcal{S}_\mathcal{R}$ could be identified through an iterative process on $\overline{\Omega}$ (rather than on the higher-dimensional $\Omega_\mathcal{U} \cup \Omega_\mathcal{S}$) [37]. So, in principle, $v(\boldsymbol{x})$ can be fully computed without increasing the dimensionality of $\overline{\Omega}$. However, to use the MFL as the "lowest budget" boundary condition for (3.21), we

also need to know the values of w_1 on the MFL. This corresponds to the "constrained optimal" cost $\tilde{u}$ along resource-optimal trajectories defined by v; see Table 1 for a similar example in the discrete setting. The function $\tilde{u}$ is formally defined below; here we note that it can also be computed in the process of solving (4.11) provided w_2 is a priori known on $\partial\mathcal{S}_{\mathcal{R}}$. This is indeed the case when no resets are possible; i.e., $\mathcal{U} = \Omega$ and $\mathcal{S}_{\mathcal{R}} \subset \mathcal{S} = \partial\Omega$, precisely the setting previously considered in [27]. Unfortunately, for the general case ($\mathcal{U} \neq \Omega$), we do not know of any (fast, lower-dimensional) algorithm to compute w_2 on $\partial\mathcal{S}_{\mathcal{R}}$. (Note that in a similar discrete example depicted in Figure 5, the values of the safe nodes continue changing until the last iteration.) Instead, we proceed to recover the values on the MFL iteratively, using values of w_2^k on $\partial\mathcal{S}_{\mathcal{R}}$.

To describe this iterative process, we first define the kth approximation of the reachable subset of $\mathcal{S}$ given by $\mathcal{S}_{\mathcal{R}}^k = \{\boldsymbol{x} \in \mathcal{S} \mid w_2^k(\boldsymbol{x}) < \infty\}$. Then, MFL^k, the kth approximation of the MFL, is a graph of $v^k(\boldsymbol{x}) = v[w_1^k](\boldsymbol{x})$, which can be computed by solving (4.11) with boundary conditions (4.10) where $\mathcal{S}_{\mathcal{R}}$ is replaced by $\mathcal{S}_{\mathcal{R}}^{k-1}$. The numerical approximation $V^k(\boldsymbol{x})$ can be efficiently computed using the methods discussed in section 4.1.

Once v^k is known, we can define the subset of $\mathcal{U}$ that is reachable at the kth iteration as $\mathcal{U}_{\mathcal{R}}^k = \left\{\boldsymbol{x} \in \mathcal{U} \mid v^k(\boldsymbol{x}) \leq B\right\}$, and a function $\tilde{u}^k \colon \mathcal{U}_{\mathcal{R}}^k \to \boldsymbol{R}$ as

$$\tilde{u}^k(\boldsymbol{x}) = \begin{cases} w_1^k(\boldsymbol{x}, v^k(\boldsymbol{x})), & \boldsymbol{x} \in \mathcal{U}_{\mathcal{R}}^k, \\ w_2^{k-1}(\boldsymbol{x}), & \boldsymbol{x} \in \partial\mathcal{U}. \end{cases} \tag{4.12}$$

Since we intend to use $\tilde{u}^k$ as a "lower boundary" condition for w_1^k, we must compute $\tilde{u}^k$ using only the information derived from w_2^{k-1}, already computed at that stage of the algorithm. For this purpose, it is possible to represent $\tilde{u}^k$ as a value function of another related control problem on $\mathcal{U}_{\mathcal{R}}^k$.

Let $T = T(\boldsymbol{x}, b, \boldsymbol{a}) = \min\{t \mid \boldsymbol{y}(t) \in \mathcal{S}_{\mathcal{R}}^{k-1}\}$. Define $\tilde{\mathcal{A}}^k(\boldsymbol{x})$ to be the set of all "v^k-optimal" controls, i.e., the controls which lead from $\boldsymbol{x}$ to $\mathcal{S}_{\mathcal{R}}^{k-1}$ through $\mathcal{U}_{\mathcal{R}}^k$ using exactly $v^k(\boldsymbol{x})$ in resources. For most starting positions $\boldsymbol{x} \in \mathcal{U}_{\mathcal{R}}^k$, this set will be a singleton, but if multiple feasible controls are available, their corresponding primary costs can be quite different. Then $\tilde{u}^k$ can be characterized as

$$\tilde{u}^k(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot) \in \tilde{\mathcal{A}}^k(\boldsymbol{x})} \int_0^T K(\boldsymbol{y}(t), \boldsymbol{a}(t))\, dt + w_2^{k-1}(\boldsymbol{y}(T)). \tag{4.13}$$

By Bellman's optimality principle, (4.13) yields

$$\tilde{u}^k(\boldsymbol{x}) = \lim_{\tau \to 0^+} \min_{\boldsymbol{a} \in A^k(\boldsymbol{x})} \{\tau K(\boldsymbol{x}, \boldsymbol{a}) + \tilde{u}^k(\boldsymbol{x} + \tau \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}))\}, \tag{4.14}$$

where $A^k(\boldsymbol{x}) \subset A$ is the set of minimizing control values in (4.11).

If $\tilde{U}^k(\boldsymbol{x})$ is the approximation to $\tilde{u}^k(\boldsymbol{x})$ at a gridpoint $\boldsymbol{x} \in \mathcal{G}_{\mathcal{U}}$, a natural semi-Lagrangian scheme based on (4.14) is

$$\tilde{U}^k(\boldsymbol{x}) = \min_{\boldsymbol{a} \in A^k(\boldsymbol{x})} \{\tau K(\boldsymbol{x}, \boldsymbol{a}) + \tilde{U}^k(\boldsymbol{x} + \tau \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}))\}, \qquad \boldsymbol{x} \in \mathcal{U}_{\mathcal{R}}^k, \tag{4.15}$$

where $\tilde{U}^k(\boldsymbol{x} + \tau \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}))$ is interpolated, and the corresponding boundary condition is

$$\tilde{U}^k(\boldsymbol{x}) = W_2^{k-1}(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathcal{G} \cap \mathcal{S}_{\mathcal{R}}^{k-1}. \tag{4.16}$$

Since the set $A^k(\boldsymbol{x})$ has to be found when solving (4.11), it is also natural to solve (4.15) at each gridpoint as soon as its V^k becomes available.

As discussed above, $\tilde{U}^k$ acts as a numerical boundary condition on the surface $b = V^k(\boldsymbol{x})$ for the update scheme (4.8). However, in general, $V^k(\boldsymbol{x}) \notin \mathcal{B}$.

In our implementation, we set $W_1^k(\boldsymbol{x}, b_j) = \tilde{U}^k(\boldsymbol{x})$, where j is the smallest integer such that $V^k(\boldsymbol{x}) \le j\Delta b$. This introduces additional $O(\Delta b)$ initialization errors at the MFL. An alternative (more accurate) approach would require using cut-cells when interpolating near the MFL.

The resulting iterative method is summarized in Algorithm 2. We note that the following properties are easy to prove inductively using the comparison principle [4] on the PDEs (3.21), (3.22), and (4.11).

Initialization:
$W_1^0(\boldsymbol{x}, b) := \infty, \quad \forall\ \boldsymbol{x} \in \mathcal{G}_{\mathcal{U}}, b \in \mathcal{B};$
$W_2^0(\boldsymbol{x}) := \infty, \quad \forall\ \boldsymbol{x} \in \mathcal{G}_{\mathcal{S}} \backslash \partial\Omega;$
$W_2^0(\boldsymbol{x}) := q(\boldsymbol{x}), \quad \forall\ \boldsymbol{x} \in \partial\Omega;$
Compute $U(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{G}$;

Main Loop:
foreach $k = 1, 2, \ldots$ *until* W_1^k *and* W_2^k *stop changing* **do**
  Using W_2^{k-1} to specify $\mathcal{G} \cap \mathcal{S}_{\mathcal{R}}^{k-1}$, compute V^k and $\tilde{U}^k$ for each $\boldsymbol{x} \in \mathcal{G}_{\mathcal{U}}$.
  Phase I:
  foreach $b = \Delta b,\ 2\Delta b, \ldots, B$ **do**
    foreach $\boldsymbol{x} \in \mathcal{G}_{\mathcal{U}}$ **do**
      if $b \ge V^k(\boldsymbol{x})$ **then**
        if $b < (V^k(\boldsymbol{x}) + \Delta b)$ **then**
          $W_1^k(\boldsymbol{x}, b) := \tilde{U}^k(\boldsymbol{x})$;
        else
          if $W_1^k(\boldsymbol{x}, b - \Delta b) = U(\boldsymbol{x})$ **then**
            $W_1^k(\boldsymbol{x}, b) := U(\boldsymbol{x})$;
          else
            Compute $W_1^k(\boldsymbol{x}, b)$ from equation (4.8);
          end
        end
      end
    end
  end
  Phase II:
  Compute W_2^k on $\mathcal{G}_{\mathcal{S}}$;
end

Algorithm 2: The budget reset problem algorithm.

We note that the following properties are easy to prove inductively using the comparison principle [4] on the PDEs (3.21), (3.22), and (4.11).

PROPOSITION 4.1. *The iterative method is monotone in the following sense:*

1. $w^{k+1}(\boldsymbol{x}, b) \leq w^k(\boldsymbol{x}, b)$ for each $(\boldsymbol{x}, b) \in \Omega_r$ and $k = 0, 1, 2, \ldots$.
2. $\mathcal{S}_{\mathcal{R}}^k \subseteq \mathcal{S}_{\mathcal{R}}^{k+1}$ for each $k = 1, 2, \ldots$.
3. $v^{k+1}(\boldsymbol{x}) \leq v^k(\boldsymbol{x})$ for each $\boldsymbol{x} \in \Omega$ and each $k = 1, 2, \ldots$.

Remark 4. We briefly discuss the convergence of W_2 and W_1 to w_2 and w_1, respectively, as $h, \Delta b \to 0$. Under the listed assumptions, the value function w_2 is Lipschitz continuous on $\text{int}(\mathcal{S})$ and can be approximated by the methods described in section 4.1; these approximations will induce errors depending on h only, since W_2 is approximated only in the top slice $b = B$. For example, standard Eulerian type (first-order upwind finite difference) discretizations are $O(h)$ accurate, provided the solution has no rarefaction-fan-type singularities on the influx part of the boundary. (The latter issue is illustrated by the convergence test in section 5.1.)

On the other hand, the value function w_1 can be discontinuous on $\Omega_{\mathcal{U}}$ and a weaker type of convergence is to be expected as a result. In the absence of resets (i.e., with $\mathcal{U} = \Omega$), if we focus on any compact $\mathcal{K} \subset \Omega_{\mathcal{U}}$ on which w_1 is continuous, then the semi-Lagrangian scheme (4.8) has been proven to converge to w_1 on $\mathcal{K}$ uniformly, provided $h = o(\Delta b)$ as $h, \Delta b \to 0$; see [5, 6]. To the best of our knowledge, there are no corresponding theoretical results for convergence to discontinuous viscosity solutions of hybrid control problems (e.g., (3.21)). Nevertheless, the numerical evidence strongly supports the convergence of described schemes (section 5.1). In [27] it was empirically demonstrated that without resets the L_1-norm convergence (or L_∞-norm convergence away from discontinuities) can often be attained even with a less restrictive choice of $h = O(\Delta b)$. In section 5.1, we show that this also holds true even if budget resets are allowed.

Finally, we note two additional sources of "lower boundary" errors: due to an approximation of the MFL and due to an approximation of $w_1 = \tilde{u}$ on it; the first of these is $O(\Delta b)$ while the latter is $O(h)$.

The optimal paths in Ω_r can be extracted from the value function in a manner similar to the description in section 3.1. The only additional computation is in the b-direction for the parts of trajectory in $\Omega_{\mathcal{U}}$; for example, in the isotropic case, the budget β along the optimal path $\boldsymbol{y}^*$ decreases by $\hat{K}(\boldsymbol{y}^*(t))$. For each $(\boldsymbol{x}, b)$, the optimal control value $\boldsymbol{a}^*$ can be found either from an approximation of $\nabla_{\boldsymbol{x}} U$ or by solving the local optimization problem similar to (4.8). Once $\boldsymbol{a}^*$ is known, the system (3.18) can be integrated forward in time by any ODE solver (our implementation uses Euler's method).

5. Numerical results. For the sake of simplicity, we will assume that the dynamics are isotropic (i.e., $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{a}) = \boldsymbol{a} f(\boldsymbol{x})$), the primary running cost is $K \equiv 1$ with the zero exit cost on the target (making the value function equal to the total time along the optimal trajectory), and $\hat{K} \equiv 1$ (constraining the maximum contiguous time spent in $\mathcal{U}$) in all of the examples.

In addition to a numerical convergence test of Algorithm 2 (section 5.1), we will illustrate the effects of geometries of $\mathcal{U}$, $\mathcal{S}$, spatial inhomogeneity of the speed f and different maximum budgets B. For each example we show the level curves of the value function at the top b-slice (i.e., $w(\boldsymbol{x}, B)$). In subsections 5.3 and 5.4, we also show constrained-optimal paths, starting at some representative point $\boldsymbol{x} \in \mathcal{U}$ with the maximum starting budget $b = B$. We emphasize that, for other starting budgets $b < B$, the constrained-optimal paths can be quite different, but all the data needed

to recover them is also a by-product of Algorithm 2.

The numerical solutions are computed on $\Omega = [-1,1]^2$ discretized by a uniform $N \times N$ Cartesian grid. In all examples except for the convergence tests in section 5.1, we use $N = 300$, $h = 2/(N-1)$, and the budget direction is discretized with $\Delta b = B/\text{round}\left(\frac{B}{0.8h}\right)$; resulting in $N_b = |\mathcal{B}| = (B/\Delta b) + 1 = O(1/h)$. The main loop in Algorithm 2 was terminated when both $\|W_1^k - W_1^{k-1}\|_\infty$ and $\|W_2^k - W_2^{k-1}\|_\infty$ fell below the tolerance threshold of 10^{-8}.

All tests were performed in MATLAB (version R2010b) with most of the implementation compiled into MEX files. The tests were conducted on a 2.6 GHz MacBook computer under Mac OS X with 4 GB of RAM. On average, the computations took approximately one minute of processing time, but we emphasize that our implementation was not optimized for maximum efficiency.

5.1. Convergence test. We test the convergence of Algorithm 2 with $\mathcal{S} = \{(x,y) \in \Omega \mid x \le 1/3\} \cup \partial\Omega$. Assume isotropic, unit speed and costs $f(\boldsymbol{x},\boldsymbol{a}) = \boldsymbol{a}$, $K = \hat{K} = 1$, $A = S^1$, with maximum budget $B = 1$. We consider the case of a point "target" $\mathcal{T} = (1,0)$ by choosing the boundary conditions

$$w(x,y,b) = \begin{cases} 0, & (x,y) = \mathcal{T}, \\ +\infty, & (x,y) \in \partial\Omega \backslash \{\mathcal{T}\}. \end{cases} \tag{5.1}$$

Note that the problem is symmetric with respect to the x axis; moreover, an explicit formula for the exact solution can be derived from simple geometric considerations.

To simplify the notation, we define a few useful geometric entities on the domain (see Figure 6 for a diagram):

$$\begin{aligned} \mathcal{L} &= \text{the vertical line segment at } x = \frac{1}{3} \text{ for } -\frac{\sqrt{5}}{3} \le y \le \frac{\sqrt{5}}{3}. \\ P_1, P_2 &= \text{the upper and lower end points of } \mathcal{L}, \text{ respectively.} \\ P(x,y) &= \begin{cases} P_1 & \text{if } y \ge 0, \\ P_2 & \text{otherwise.} \end{cases} \end{aligned} \tag{5.2}$$

We shall only describe an optimal (feasible) path from an arbitrary point $\boldsymbol{x} = (x,y)$ to $\mathcal{T}$, since $w(x,y,b)$ is simply the length of that path. For convenience we will use the notation "$\boldsymbol{x} \rightharpoonup \boldsymbol{y}$" as a shorthand for "a (directed) straight line segment from $\boldsymbol{x}$ to $\boldsymbol{y}$".

We begin by describing the optimal path from $\boldsymbol{x} \in \mathcal{S}$. If $\boldsymbol{x} \rightharpoonup \mathcal{T}$ passes through $\mathcal{L}$, this line segment is precisely the optimal path. If the line does not pass through $\mathcal{L}$, the optimal path is $\boldsymbol{x} \rightharpoonup P(\boldsymbol{x}) \rightharpoonup \mathcal{T}$. Next, we describe the optimal path from $\boldsymbol{x} \in \mathcal{U}$ with initial budget b. Clearly, if $\boldsymbol{x} \in \mathcal{U}$ is more than b distance from both $x = \frac{1}{3}$ and $\mathcal{T}$, then $w(\boldsymbol{x},b) = +\infty$. Also, if $\boldsymbol{x}$ is within b distance from $\mathcal{T}$, the optimal path is $\boldsymbol{x} \rightharpoonup \mathcal{T}$. Otherwise, the optimal path will have to first visit $\mathcal{S}$ (and reset the budget to B), before moving to $\mathcal{T}$. This situation can be further divided into two cases:

Case 1. If $\boldsymbol{x} \in \mathcal{U}$ is within b distance from $\mathcal{L}$, the optimal path is to move from $\boldsymbol{x} \rightharpoonup \boldsymbol{y} \rightharpoonup \mathcal{T}$, where $\boldsymbol{y}$ is a point on $\mathcal{L}$ such that $\|\boldsymbol{x} - \boldsymbol{y}\| \le b$ minimizing $\|\boldsymbol{x} - \boldsymbol{y}\| + \|\boldsymbol{y} - \mathcal{T}\|$.

Case 2. The optimal path is $\boldsymbol{x} \rightharpoonup \boldsymbol{z} \rightharpoonup P(\boldsymbol{x}) \rightharpoonup \mathcal{T}$, where $\boldsymbol{z}$ is the closest point on the line $x = \frac{1}{3}$ to $P(\boldsymbol{x})$ such that $\|\boldsymbol{x} - \boldsymbol{z}\| \le b$.

From the above, it should be clear that for each $b > 0$ the value function will have a discontinuous jump on $\mathcal{D}(b) = \{(\boldsymbol{x},b) \in \Omega_e \mid \boldsymbol{x} \in \mathcal{U},\ \|\boldsymbol{x} - \mathcal{T}\| = b\}$.

We compare the numerically computed solution to the exact solution in the L_1 and L_∞ norms. For the L_∞ norm, we compare the solutions on a subset $\Omega_e^\epsilon \subset \Omega_e$ where the w is known to be continuous:

$$\Omega_e^\epsilon = \{(\boldsymbol{x}, b) \in \Omega_e \mid \|\boldsymbol{x} - \boldsymbol{y}\| > \epsilon, \quad \forall (\boldsymbol{y}, b) \in \mathcal{D}(b)\}. \tag{5.3}$$

In particular we investigate the L_∞ norm errors for $\epsilon = \epsilon(h) = 3h$ and $\epsilon = 0.1$ (independent of h). The L_1 errors are computed over the whole computational domain.

The errors are reported in Table 2. A contour plot of the numerical solution on the top b-slice is shown in Figure 6.

The convergence observed in Table 2 is actually stronger than predicted in theory. First, in this numerical test we always chose $\Delta b = h$, whereas the theory (even for the no resets case) guarantees convergence for $h = o(\Delta b)$ only; see Remark 4. Second, the L_∞-norm convergence is guaranteed on any fixed compact set away from discontinuity, but the choice of $\epsilon = 3h$ goes beyond that.

At the same time, the observed rate of convergence (in all norms) is less than one despite our use of the first-order accurate discretization. This is not related to any discontinuities in the solution, but is rather due to the "rarefaction fans" (characteristics spreading from a single point) present in this problem. Indeed, this phenomenon is well known even for computations of distance function from a single target point: a cone-type singularity in the solution results in much larger local truncation errors near the target, thus lowering the rate of convergence. A "singularity factoring" approach recently introduced in [22] allows one to circumvent this issue at the target, but we note that there are two more rarefaction fans spreading from points P_1 and P_2; see Figure 6. (Intuitively, this is due to the fact that optimal trajectories from infinitely many starting points pass through P_1 or P_2 on their way to $\mathcal{T}$.) Since, in general examples, the locations and types of such rarefaction fans are a priori unknown, "factoring methods" similar to those in [22] are not applicable.

5.2. Geometry of $\mathcal{S}$ and the number of iterations. We illustrate how the information propagates within the main loop of Algorithm 2. Since the reachable part of the safe set ($\mathcal{S}_\mathcal{R}$) is obtained iteratively, it might seem natural to expect that the iterative process stops once all the reachable components of $\mathcal{S}$ are already found (i.e., once $\mathcal{S}_\mathcal{R}^k = \mathcal{S}_\mathcal{R}$, also ensuring that $MFL^k = MFL$). Here we show that this generally need not be the case and the value function W^k might require more iterations to converge after that point. Roughly speaking, this occurs when the order of traversal of some optimal path through a sequence of connected components of $\mathcal{S}_\mathcal{R}$ differs from the order in which those components were discovered. Mathematically, the extra iterations are needed because the correct values of W_2^k are not yet known on Γ, and the values of W_1^{k+1} on the MFL are still incorrect as a result.

Consider the following "pathological" example (shown in Figure 7): $\mathcal{S}$ consists of eight square blocks $S_1, S_2, \ldots, S_8$ with side lengths 0.4, enumerated counterclockwise,

TABLE 2
Errors measured against the exact solution for various grid sizes N.

N	h	$L_1(\Omega_e)$	Rate	$L_\infty(\Omega_e^\epsilon)$, $\epsilon = 3h$	Rate	$L_\infty(\Omega_e^\epsilon)$, $\epsilon = 0.1$	Rate
61	1/30	0.0612	-	0.0681	-	0.0681	-
121	1/60	0.0309	0.99	0.0513	0.41	0.0416	0.71
241	1/120	0.0151	1.03	0.0401	0.36	0.0255	0.71
481	1/240	0.0076	0.99	0.0265	0.60	0.0151	0.76
961	1/480	0.0039	0.96	0.0162	0.71	0.0090	0.75

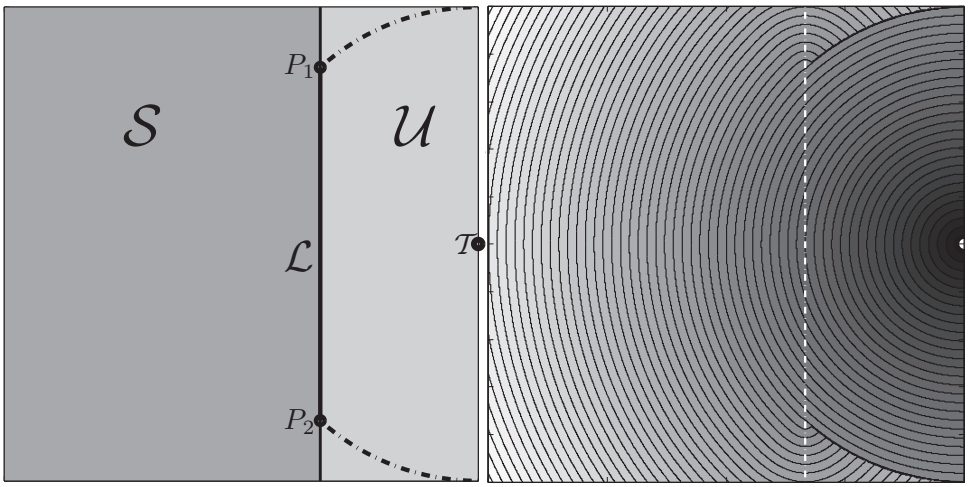


FIG. 6. *Left: an illustration of the domain. The darker region is $\mathcal{S}$ and the lighter region is $\mathcal{U}$. The target is $\mathcal{T} = (1, 0)$. The black dotted line is the circle of radius $B = 1$ centered about $\mathcal{T}$ in $\mathcal{U}$. Right: a contour plot of the numerical solution of $w(x, y, 1)$ for grid size $N = 961$. The vertical white dotted line is the interface between $\mathcal{S}$ and $\mathcal{U}$ at $x = 1/3$.*

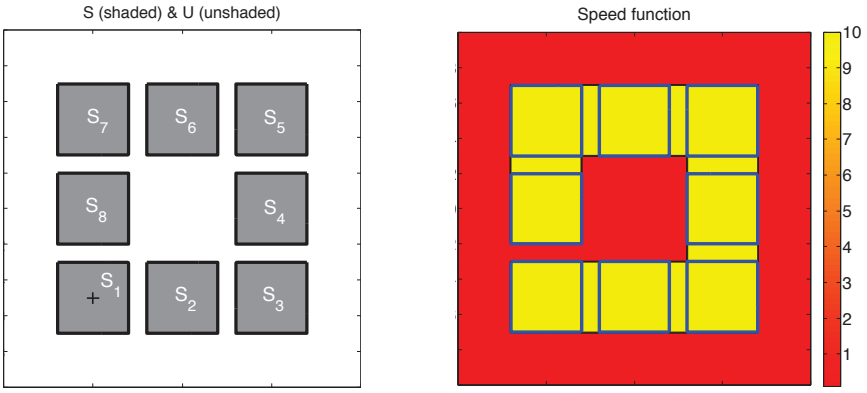


FIG. 7. *Left: The sets $\mathcal{S}$, $\mathcal{U}$ and the target $\mathcal{T} = (-0.5, -0.5)$ shown by a black cross. The eight connected components $S_1, S_2, \ldots, S_8$ of $\mathcal{S}$ are labeled. Right: the speed function. Note the slow speed in the corridor C_8 between the S_1 and S_8.*

with $\mathcal{T} \in S_1$. To simplify the discussion, we assume that $S_9 := S_1$ and introduce the notation for "unsafe corridors" between the safe squares:

$$C_i \;=\; \text{convex\_hull}(S_i \cup S_{i+1}) \cap \mathcal{U}, \qquad i = 1, \ldots, 8.$$

We set

$$f(\boldsymbol{x}) = \begin{cases} 10, & \boldsymbol{x} \in \mathcal{S} \cup C_1 \cup C_2 \cup \cdots \cup C_7, \\ 0.1 & \text{otherwise.} \end{cases}$$

Note that $f = 0.1$ in C_8. The target is $\mathcal{T} = (-0.5, -0.5)$ and the maximum budget is $B = 1.5$. Note that for all the starting positions in $S_2, \ldots, S_8$, the corresponding constrained optimal trajectories will run toward S_1 clockwise (to avoid

crossing through the slow region in C_8). The same is true for all starting positions in $C_1, \ldots, C_7$ and even for points in C_8 that are significantly closer to S_8.

This problem was specifically designed to illustrate the difference between the evolution of the reachable set $\Omega^k_{\mathcal{R}} = \mathcal{U}^k_{\mathcal{R}} \cup \mathcal{S}^k_{\mathcal{R}}$ and the evolution of the "fully converged" set $\mathcal{F}^k = \{\boldsymbol{x} \mid w^k(\boldsymbol{x}) = w(\boldsymbol{x})\}$, on which the value function is correct after the kth iteration. Both sets eventually contain all S_i's and C_i's, but a careful examination of Figure 8 reveals the difference. For the reachable set, $\Omega^1_{\mathcal{R}} = S_1$ initially, and the algorithm iteratively discovers the reachable S_i's (and C_i's) simultaneously in both the clockwise and counterclockwise directions. More than one S_i can be "discovered" per iteration in each direction; e.g., at iteration $k = 3$, Phase I of the algorithm discovers that $C_2 \cup C_3 \subset \mathcal{U}^3_{\mathcal{R}}$ owing to a feasible path that "passes through the corner" shared by C_2 and C_3; this subsequently leads to Phase II discovering that $S_3 \cup S_4 \subset \mathcal{S}^3_{\mathcal{R}}$. The same argument implies that $C_6 \cup C_7 \subset \mathcal{U}^3_{\mathcal{R}}$ and $S_6 \cup S_7 \subset \mathcal{S}^3_{\mathcal{R}}$. After one more iteration we already see that $S_5 \subset \mathcal{S}^4_{\mathcal{R}}$ and another iteration is sufficient to recover the entire reachable set $\Omega_{\mathcal{R}} = \Omega^5_{\mathcal{R}}$.

However, on a large part of $\Omega_{\mathcal{R}}$ the value function is still quite far from correct at that point; e.g., a comparison of level curves in C_5 and S_5 shows that $C_5 \notin \mathcal{F}^5$. Since $\mathcal{T} \in S_1 \subset \mathcal{S}$ and S_1 is convex, we have $S_1 \subset \mathcal{F}^1$. It is similarly easy to see that $(\mathcal{F}^{k-1} \cup C_{k-1} \cup S_k) \subset \mathcal{F}^k$ for $k = 2, \ldots, 8$. Thus, it takes eight iterations to correctly compute w on the entire safe set and one more iteration to cover those unsafe points (including a part of C_8), whose optimal trajectories take them first to S_8 and then clockwise (through the "fast belt") toward S_1. We note that, as iterations progress, the value function need not be converged on recently discovered components of $\mathcal{S}_{\mathcal{R}}$, even if the level sets already show the generally correct (clockwise) direction of optimal trajectories. For example, it might seem that $S_8 \in \mathcal{F}^6$, but a careful comparison of level curves shows that the value function is still incorrect even on C_6 and S_7. (This is due to the fact that the reachability of S_8 is discovered by feasible trajectories passing through a common corner of C_6 and C_7.) Figure 9 confirms this by showing the ∞-norm of the value changes after each iteration. We observe two key events:

- the initial drop in value changes when $\Omega_{\mathcal{R}}$ is fully discovered after iteration five (at which point the errors are still independent of h and Δb);

- and the convergence (i.e., the drop of value changes to the machine precision) after iteration nine.

5.3. Optimal paths and the effect of varying B. We now consider two examples with inhomogeneous speed functions. The first scenario involves a discontinuous speed function that is slow in the safe set:

$$f(\boldsymbol{x}) = \begin{cases} 1, & \boldsymbol{x} \in \mathcal{U}, \\ 0.3, & \boldsymbol{x} \in \mathcal{S}. \end{cases}$$

This example presents a curious dilemma: any good path will try to avoid $\mathcal{S}$ to travel faster to $\mathcal{T}$, but it must visit $\mathcal{S}$ at least every B distance to keep the budget from depleting. The numerical test for $B = 0.4$ is shown in the center plot of Figure 10. The computed "optimal path" tends to travel along the interface Γ on the $\mathcal{U}$ side while occasionally making short visits into $\mathcal{S}$ to reset the budget.[4] We have added

[4]Since $\Gamma \subset \mathcal{S}$, it is not really possible to quickly travel on the Γ itself. As a result, an optimal control does not exist, though the value function w_1 is still well-defined. This lack of optimal

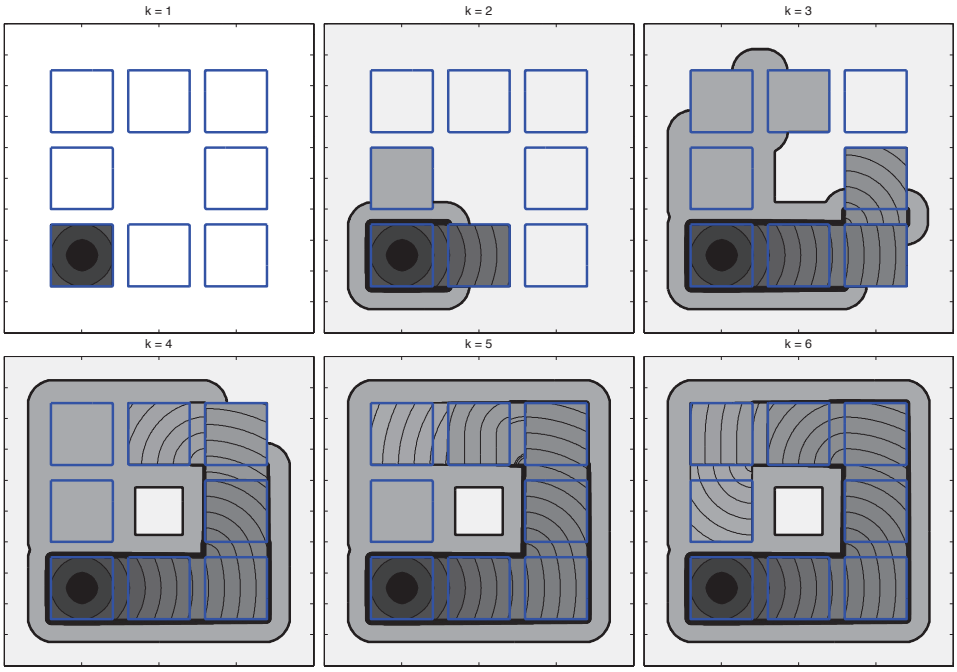


FIG. 8. *The first six iterations (each after Phase* II*) of* W^k *on* $b = B$*. The contour lines are scaled logarithmically to avoid bunching in the slow regions of* $\mathcal{U}$*.*

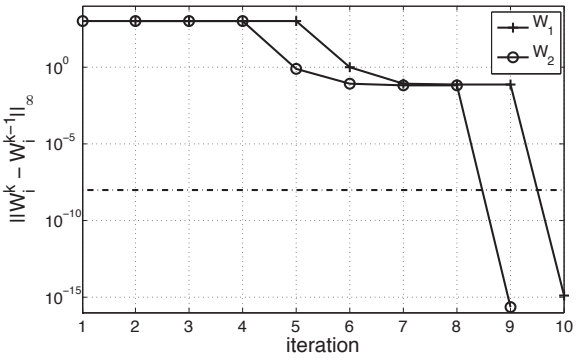


FIG. 9. *The max-norm difference between consecutive* W_1^k *and* W_2^k*. The numerical convergence tolerance is shown by the thick horizontal dotted line. Both* W_1^0 *and* W_2^0 *were set to* 10^3*.*

small white circles to the plot in Figure 10 (center) to identify the locations of these "reentry" points.

The second example illustrates the robustness of the numerical scheme to non-trivial speed functions in $\mathcal{U}$: we set $B = 0.4$ and

$$f(\boldsymbol{x}) = 1 - 0.5\sin(5\pi x)\sin(5\pi y).$$

control does not contradict the compatibility condition (3.24) (i.e., $w_1 = w_2$ on Γ). The numerically recovered "optimal" trajectory shown in Figure 10 is actually "ε-suboptimal," where $\epsilon \to 0$ under the grid refinement.

The computed value function and a sample path are shown in the right plot of Figure 10.

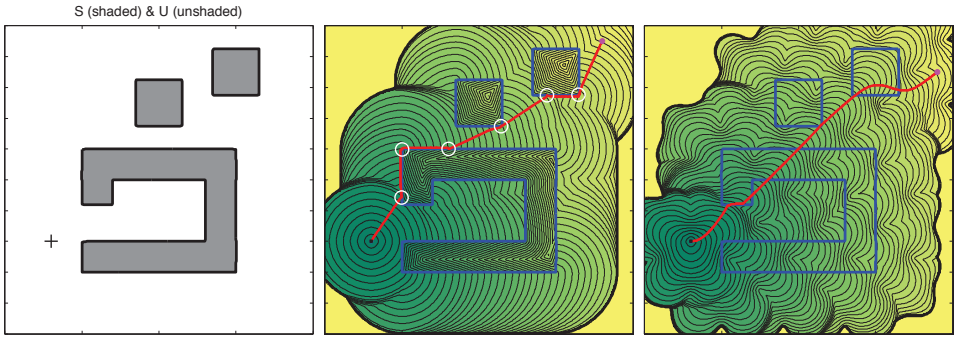


FIG. 10. *Sample optimal paths on the "islands" example with inhomogeneous speed functions.*

Next, we give numerical examples showing the effects of varying B. Figure 11 illustrates these effects on the "islands" examples (as in Figure 10). The speeds were set to $f = 1$ on all of the domains. While the optimal path is computed from the same initial point $(0.8, 0.5)$, note the large changes in its global behavior.

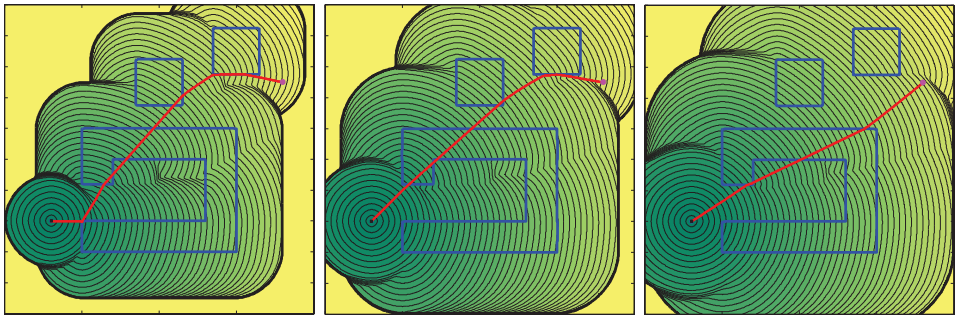


FIG. 11. *Sample optimal paths on the "islands" example for varying B. Left to right: $B = 0.3$, 0.4 and 0.5, respectively.*

5.4. Constrained contiguous visibility time. We apply Algorithm 2 to a problem involving visibility exposure: suppose the objective is to move a robot towards a target in the shortest time, among opaque and impenetrable obstacles, while avoiding "prolonged exposure" to a static enemy observer. In our budget reset problem setting, we impose the prolonged exposure constraint by letting the enemy-visible region be $\mathcal{U}$ and the nonvisible region be $\mathcal{S}$. This is similar to the problem considered in [27], except that once the robot enters the nonvisible region, it is again allowed to travel through the visible region up to the time B.

The domain consists of four obstacles which act both as state constraints and as occluders. The static observer is placed at $(0.8, 0.8)$, and the corresponding visible set is computed by solving an auxiliary (static and linear) PDE on Ω [38]. We compute the value function and optimal paths for the same starting location but two different exposure budgets: $B = 0.15$ and $B = 0.3$; see Figure 12. Note that, for small B, the budget is insufficient to directly travel across the $\mathcal{U}$ "corridor" between the

"shadows" of the larger foreground obstacles; for the larger B this shortcut is feasible, thus reducing the path length.

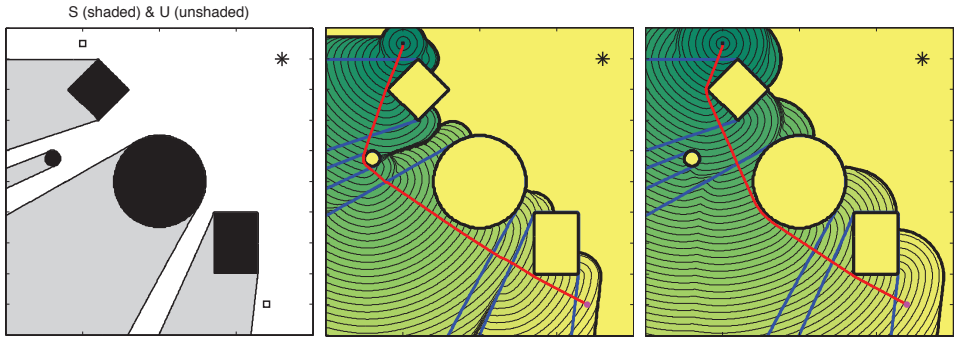


FIG. 12. *The problem of constrained contiguous visibility time. The static observer location is shown by an asterisk. Left: the observer-viewable region is white, the occluders/obstacles are black, and their "shadows" are gray. The objective of is to find the quickest path connecting the two small squares while avoiding prolonged enemy-exposure. Center and right: contour plots of* W *at* $b = B$ *and the constrained optimal paths with* $B = 0.15$ *and* 0.3*, respectively.*

6. Conclusions. In this paper we focused on computational methods for optimal control of budget-constrained problems with resets. In the deterministic case on graphs, we explained how such problems can be solved by noniterative (label-setting) methods. We then introduced new fast iterative methods, suitable for problems on graphs as well as for continuous deterministic budget reset problems. Throughout, we utilized the causal properties of the value function to make sure that dynamic programming equations are solved efficiently on the new extended domain. Connections to stochastic shortest path problems on graphs are also highlighted in the expanded version of this manuscript [37].

We presented empirical evidence of convergence and illustrated other properties of our methods on several examples, including path-planning under constraints on "prolonged exposure" to an enemy observer. Even though all selected examples are isotropic in cost and dynamics, only minor modifications (with no performance penalties) are needed to treat anisotropy in secondary cost $\hat{K}$. Anisotropies in primary cost and/or dynamics can also be easily treated by switching to a different (noniterative or fast iterative) method on the safe set $\mathcal{S}$. Several other natural extensions are likely to be more computationally expensive, but can be handled in the same framework:

- In some applications the resource restoration is more realistically modeled not as an instantaneous reset, but as a continuous process on $\mathcal{S}$. Alternatively, resets might be also modeled as conscious/optional control decisions available on $\mathcal{S}$, with an instantaneous penalty in primary cost.
- Differential games can be similarly modified to account for limited (and possibly renewable) resource budgets.
- More generally, both the dynamics and the budget changes might be affected by some random events, leading to stochastic trajectories in the extended domain.
- It would be interesting to extend the method to problems with constraints on multiple reset-renewable resources.

For problems with a fixed starting position, significant computational savings could be attained by adopting A*-type domain restriction algorithms [14]. All of these

extensions are of obvious practical importance in realistic applications, and we hope to address them in the future. Finally, more work is clearly needed to provide proofs of convergence of semi-Lagrangian schemes to discontinuous viscosity solutions of hybrid systems.

Acknowledgments. The authors would like to thank Professor R. Tsai and Dr. Y. Landa, whose work on visibility and surveillance-evasion problems (joint with Ryo Takei) served as a starting point for this paper. The authors are grateful to them for helping to formulate the problem in section 5.4 and for many motivating discussions. The authors are also grateful to Ajeet Kumar, whose source code developed in [27] served as a starting point for our implementation described in section 3. RT would also like to thank Professors S. Osher and C. Tomlin for their encouragement and hospitality during the course of this work.

REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows. Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, NJ, 1993.

[2] K. Alton and I. M. Mitchell, *An ordered upwind method with precomputed stencil and monotone node acceptance for solving static convex Hamilton-Jacobi equations*, J. Sci. Comput., 51 (2012), pp. 313–348.

[3] M. Bardi and M. Falcone, *An approximation scheme for the minimum time function*, SIAM J. Control Optim., 28 (1990), pp. 950–965.

[4] M. Bardi and I. Capuzzo Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, Birkhäuser Boston, Boston, 1997.

[5] M. Bardi, M. Falcone, and P. Soravia, *Fully discrete schemes for the value function of pursuit-evasion games*, Advances in Dynamic Games and Applications, T. Basar and A. Haurie, eds., Birkhäuser Boston, Boston, 1994, pp. 89–105.

[6] M. Bardi, M. Falcone, and P. Soravia, *Numerical methods for pursuit-evasion games via viscosity solutions,* Stochastic and Differential Games, M. Bardi, T. E. S. Raghavan, and T. Parthasarathy, eds., Birkhäuser Boston, Boston, 1999, pp. 105–175.

[7] R. Bellman, *On the theory of dynamic programming*, Proc. Natl. Acad. Sci. USA, 38 (1952), pp. 716–719.

[8] D. P. Bertsekas, *Network Optimization: Continuous & Discrete Models*, Athena Scientific, Boston, 1998.

[9] M. Boué and P. Dupuis, *Markov chain approximations for deterministic control problems with affine dynamics and quadratic cost in the control*, SIAM J. Numer. Anal., 36 (1999), pp. 667–695.

[10] S. Cacace, E. Cristiani, and M. Falcone, *A local ordered upwind method for Hamilton-Jacobi and Isaacs equations*, in Proceedings of the 18th IFAC World Congress, 2011, pp. 6800–6805.

[11] E. Carlini, M. Falcone, and R. Ferretti, *A time-adaptive semi-Lagrangian approximation to mean curvature motion*, Numerical Mathematics Advanced Applications- ENUMATH2005, Springer, Berlin, 2006, pp. 732–739.

[12] E. Carlini, M. Falcone, N. Forcadel, and R. Monneau, *Convergence of a generalized fast-marching method for an eikonal equation with a velocity-changing sign*, SIAM J. Numer. Anal., 46 (2008), pp. 2920–2952.

[13] A. Chacon and A. Vladimirsky, *Fast two-scale methods for eikonal equations*, SIAM J. Sci. Comput., 34 (2012), pp. A547–A578.

[14] Z. Clawson, A. Chacon, and A. Vladimirsky, *Causal domain restriction for eikonal equations*, SIAM J. Sci. Comput., 36 (2014), pp. A2478–A2505.

[15] M. G. Crandall and P.-L. Lions, *Viscosity Solutions of Hamilton-Jacobi Equations*, Trans. Amer. Math. Soc., 277 (1983), pp. 1–42.

[16] E. Cristiani and M. Falcone, *A Fast Marching Method for Pursuit-Evasion Games*, published electronically in "Communications to SIMAI Congress", SIMAI 2006 (Baia Samuele, Ragusa, Italy, 2006), Vol. 1, 2006.

[17] R. Dial, *Algorithm* 360: *Shortest path forest with topological ordering*, Comm. ACM, 12 (1969), pp. 632–633.

[18] E. W. Dijkstra, *A note on two problems in connection with graphs*, Numer. Math., 1 (1959), pp. 269–271.
[19] M. Falcone, *The minimum time problem and its applications to front propagation*, in Motion by Mean Curvature and Related Topics, Proceedings of the International Conference at Trento, 1992, de Gruyter, Berlin, 1994, pp. 70–88.
[20] M. Falcone, *A numerical approach to the infinite horizon problem of deterministic control theory*, Appl. Math. Optim., 15 (1987), pp. 1–13; corrigenda 23 (1991), pp. 213–214.
[21] M. Falcone and R. Ferretti, *Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations*, Numer. Math., 67 (1994), pp. 315–344.
[22] S. Fomel, S. Luo, and H. Zhao, *Fast sweeping method for the factored eikonal equation*, J. Comput. Phys. 228 (2009), pp. 6440–6455.
[23] R. Gonzalez and E. Rofman, *On deterministic control problems: An approximate procedure for the optimal cost* I. *The stationary problem*, SIAM J. Control Optim., 23 (1985), pp. 242–266.
[24] P. Hansen, *Bicriterion path problems*, in Multiple Criteria Decision Making: Theory and Applications, G. Fandel and T. Gal, eds., Springer, Berlin, New York, 1980, pp. 109–127.
[25] J. M. Jaffe, *Algorithms for finding paths with multiple constraints*, Networks, 14 (1984), pp. 95–116.
[26] C. Y. Kao, S. Osher, and J. Qian, *Lax-Friedrichs sweeping scheme for static Hamilton-Jacobi equations*, J. Comput. Phys., 196 (2004), pp. 367–391.
[27] A. Kumar and A. Vladimirsky, *An efficient method for multiobjective optimal control and optimal control subject to integral constraints*, J. Comput. Math., 28 (2010), pp. 517–551.
[28] H. J. Kushner and P. G. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time*, Springer-Verlag, New York, 1992.
[29] E. Q. V. Martins, *On a multicriteria shortest path problem*, European J. Oper. Res., 16 (1984), pp. 236–245.
[30] M. Motta and F. Rampazzo, *Multivalued dynamics on a closed domain with absorbing boundary. Applications to optimal control problems with integral constraints*, Nonlinear Anal., 41 (2000), pp. 631–647.
[31] J. A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. USA, 93 (1996), pp. 1591–1595.
[32] J. A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Sciences*, Cambridge University Press, Cambridge, UK, 1996.
[33] J. A. Sethian, *Fast marching methods*, SIAM Rev., 41 (1999), pp. 199–235.
[34] J. A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations*, Proc. Natl. Acad. Sci. USA, 98 (2001), pp. 11069–11074.
[35] J. A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton–Jacobi equations: Theory and algorithms*, SIAM J. Numer. Anal., 41 (2003), pp. 325–363.
[36] P. Soravia, *Viscosity solutions and optimal control problems with integral constraints*, Systems Control Lett., 40 (2000), pp. 325–335.
[37] R. Takei, W. Chen, Z. Clawson, S. Kirov, and A. Vladimirsky, *Optimal control with reset-renewable resources*, Technical report; preprint available from http://arxiv.org/abs/1110.6221.
[38] Y.-H. R. Tsai L.-T. Cheng, S. Osher, P. Burchard, and G. Sapiro, *Visibility and its dynamics in a PDE based implicit framework*, J. Comput. Phys., 199 (2004), pp. 260–290.
[39] Y.-H. R. Tsai, L.-T. Cheng, S. Osher, and H.-K. Zhao, *Fast sweeping algorithms for a class of Hamilton–Jacobi equations*, SIAM J. Numer. Anal., 41 (2003), pp. 673–694.
[40] J. N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Trans. Automat. Control, 40 (1995), pp. 1528–1538.
[41] A. Vladimirsky, *Label-setting methods for multimode stochastic shortest path problems on graphs*, Math. Oper. Res., 33 (2008), pp. 821–838.
[42] H. K. Zhao, *Fast sweeping method for eikonal equations*, Math. Comp., 74 (2005), pp. 603–627.