A Minimum Discounted Reward Hamilton-Jacobi Formulation for Computing Reachable Sets

Anayo K. Akametalu[1], Shromona Ghosh[1], Jaime F. Fisac[1], and Claire J. Tomlin[1]

Abstract—**We propose a novel formulation for approximating reachable sets through a minimum discounted reward optimal control problem. The formulation yields a continuous solution that can be obtained by solving a Hamilton-Jacobi equation. Furthermore, the numerical approximation to this solution can be obtained as the unique fixed-point to a contraction mapping. This allows for more efficient solution methods that could not be applied under traditional formulations for solving reachable sets. In addition, this formulation provides a link between reinforcement learning and learning reachable sets for systems with unknown dynamics, allowing algorithms from the former to be applied to the latter. We use two benchmark examples, double integrator, and pursuit-evasion games, to show the correctness of the formulation as well as its strengths in comparison to previous work.**

I. INTRODUCTION

Computing reachable sets has been a popular approach for verifying and validating the behavior of dynamical systems. In the reachability problem, one specifies a target set in the state space and then aims to find the set of initial states admitting trajectories that can eventually enter the target within some specified time horizon. The target may represent a region we desire to drive the system to or keep the system away from. Due to its generality, Hamilton-Jacobi (HJ) reachability analysis, which casts the problem in the optimal control framework, has seen broad usage in many safety-critical applications including emergency landing of unmanned aerial vehicles (UAVs) [14], vehicle platooning [10], safe learning [1], [19], collision-avoidance [20], [24], and many others [13], [22].

The standard HJ formulation for reachability solves a *minimum reward* (MR) optimal control problem, by computing the viscosity solution of a particular time-dependent HJ partial differential equation (PDE) [24]. Like most optimal control problems, the solution is solved approximately on a grid via value iteration, which recursively applies a *backup operator* until convergence. One downside of MR optimal control problems is that the backup operator in this setting is not

a contraction mapping, thus a specific initialization of value iteration is required for convergence to the correct solution. A contraction mapping allows for arbitrary initialization, and with a good initialization convergence can be accelerated. This is particularly useful when computing reachable sets for multiple systems with similar dynamics allowing for the solution of one problem to be used as an initialization for the other.

On the other hand, infinite horizon *sum of discounted reward* (SDR) problems yield backup operators that are contraction mappings [8]. This allows for more efficient solution methods, like policy iteration [21], [27] and multigrid approaches [3], [11].

Drawing inspiration from SDR problems, we propose a *minimum discounted reward* (MDR) formulation for computing tight over- and under-approximations of infinite horizon reachable sets. This new formulation yields a contraction mapping, making it possible to extend policy iteration and multigrid approaches to this setting. In addition, it also provides a way forward for learning reachable sets when dynamics are unknown or difficult to model, a topic that has garnered some attention recently [2], [15]. This draws greatly from Reinforcement Learning (RL), which attempts to solve the infinite horizon SDR problem when the system model is unknown. Many RL algorithms boil down to finding the fixed-point of the backup operator associated with SDR. The techniques developed in RL, like temporal differencing [30] and Q-learning [31], can naturally be extended to do the same in the MDR setting, thus facilitating research in learning reachable sets.

The paper is organized as follows. In Section II we briefly go over the MR formulation for HJ reachability analysis, and SDR problems. In Section III we describe the MDR formulation, and prove some relevant results. In Section IV we explore policy iteration and multigrid approaches within the context of the MDR. Section V draws inspiration from RL and presents some preliminary ideas on how the formulation can be used for learning reachable sets. Section VI contains examples demonstrating the ideas developed throughout the paper, and we conclude the paper in Section VII.

Here, both the one-player and adversarial two-player settings are considered, and we opt to use the terminology from optimal control for consistency. For example *optimal control* will be used to refer to both settings, and we use terms like *reward* instead of *payoff* throughout.[1]

[1]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley. Cory Hall, Berkeley, CA 94720, United States.
{kakametalu, shromona.ghosh, jfisac, tomlin} @eecs.berkeley.edu

---

[1]Typically the terms *optimal control* and *reward* are reserved for the one player setting and *zero sum differential game* and *payoff* are the analogous terms in the adversarial two-player setting.

II. BACKGROUND

We consider a two-player optimal control problem where the first player (control) wants to keep the system away from the target set for a given time horizon and the second player (disturbance) has the opposite objective.[2] Our goal is to characterize the *infinite-horizon backward reachable set*, which is the set of initial states for which the disturbance wins the game. Going forward all mentions of reachable set refer to the infinite horizon backwards reachable set.

A. System Model

The analysis in this paper considers a fully observable system whose underlying dynamics may be non-deterministic, but bounded. We can formalize this as a dynamical system with state $x \in \mathbb{R}^n$, and two inputs, $u \in \mathcal{U} \subset \mathbb{R}^{n_u}, d \in \mathcal{D} \subset \mathbb{R}^{n_d}$ (with $\mathcal{U}$ and $\mathcal{D}$ compact) which we will refer to as the *controller* and the *disturbance*:

$$\dot{x} = f(x, u, d). \tag{1}$$

The flow field $f : \mathbb{R}^n \times \mathcal{U} \times \mathcal{D} \to \mathbb{R}^n$ is assumed to be Lipschitz continuous and bounded. In the single-player case we drop the disturbance input, and just have $f(x, u) : \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}^n$.

Letting $\mathbb{U}$ and $\mathbb{D}$ denote the collections of measurable[3] functions $u : [0, \infty) \to \mathcal{U}$ and $d : [0, \infty) \to \mathcal{D}$ respectively, and allowing the controller and disturbance to choose any such signals, the evolution of the system from any initial state x is determined (see for example [12], Ch. 2, Theorems 1.1, 2.1) by the unique continuous trajectory $\xi : [0, \infty) \to \mathbb{R}^n$ solving

$$\dot{\xi}(s) = f(\xi(s), u(s), d(s)), \text{ a.e. } s \geqslant 0,$$
$$\xi(0) = x. \tag{2}$$

Note that this is a solution in Carathéodory's *extended sense*, that is, it satisfies the differential equation *almost everywhere* (i.e. except on a subset of Lebesgue measure zero).

Throughout our analysis, we will use the notation $\xi_x^{u,d}(\cdot)$ to denote the state trajectory $t \mapsto x$ corresponding to the initial condition $x \in \mathbb{R}^n$, the control signal $u \in \mathbb{U}$ and the disturbance signal $d \in \mathbb{D}$.

B. HJ Reachability: Minimum Distance to Target

The target set $\mathcal{T}$ can be implicitly characterized as the sub-zero level set of a Lipschitz *surface function* $l : \mathbb{R}^n \to \mathbb{R}$:

$$x \in \mathcal{T} \iff l(x) < 0. \tag{3}$$

This function always exists, since we can simply choose the *signed distance function* to $\mathcal{T}$, $s_\mathcal{T}(x)$, which is Lipschitz continuous by construction.[4] We use a clipped signed distance

since the problem is solved over a fixed domain in practice, $l(x) = \min(\max(s_\mathcal{T}(x), -L), L)$ with $L > 0$, where L is usually taken to be the largest value (in magnitude) on the domain.

To express whether a given trajectory *ever* enters the target set, let the functional $\mathcal{V} : \mathbb{R}^n \times \mathbb{U} \times \mathbb{D} \to \mathbb{R}$ assign to each initial state x and input signals $u(\cdot), d(\cdot)$ the lowest value of $l(\cdot)$ achieved by trajectory $\xi_x^{u,d}(\cdot)$ over all times $t \geqslant 0$:

$$\mathcal{V}(x, u(\cdot), d(\cdot)) := \inf_{t \geqslant 0} l(\xi_x^{u,d}(t)). \tag{4}$$

This outcome $\mathcal{V}$, also referred to here as the *minimum of rewards*[5], will be nonpositive if there exists any $t \in [0, \infty)$ at which the trajectory enters the target set, and will be strictly positive if the system avoids the target for all $t \geqslant 0$.

The reachable set can then be obtained by solving a game between the controller and disturbance where the disturbance can use *nonanticipative strategies* to respond to the controller's signal. The disturbance's set of nonanticipative strategies is $\mathcal{B} = \{\beta : \mathbb{U} \to \mathbb{D} \mid \forall t \geqslant 0, \forall u(\cdot), \hat{u}(\cdot) \in \mathbb{U}, (u(\tau) = \hat{u}(\tau) \text{ a.e.} \tau \geqslant 0) \Rightarrow (\beta[u](\tau) = \beta[\hat{u}](\tau) \text{ a.e.} \tau \geqslant 0)\}$. Intuitively, the disturbance is given an instantaneous informational advantage in the game, but its actions at any given time can only depend on what the controller has done up to that time. With this in place we can define the *value function* $V(x)$ and ultimately the reachable set $\mathcal{R}(\mathcal{T})$.

$$V(x) = \inf_{\beta[u](\cdot) \in \mathcal{B}} \sup_{u \in \mathbb{U}} \mathcal{V}(x, u(\cdot), \beta[u](\cdot)), \tag{5}$$

$$\mathcal{R}(\mathcal{T}) = \{x \mid V(x) \leqslant 0\}. \tag{6}$$

The optimization in (5) is referred to here as the minimum reward optimal control problem. It has been shown that the value function for problems with outcome given by (4) can be characterized as the unique viscosity solution to a particular variational inequality [6], [7] as shown in [18]. When the problem is solved over a finite time interval $[0, T]$, the *finite-horizon value function* $V(x, t)$ can be computed by solving the HJ equation:

$$0 = \min \left\{ l(x) - V(x, t), \frac{\partial V}{\partial t}(x, t) + \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} \frac{\partial V}{\partial x}(x, t) f(x, u, d) \right\},$$
$$V(x, T) = l(x). \tag{7}$$

As $T \to \infty$, $V(x, t)$ becomes independent of t. We accordingly drop the dependence on t and recover $V(x)$ as defined in (5).

C. Computing the Value Function

Several approximation schemes have been proposed for solving (7) and similar HJ equations on a fixed grid G [5], [17], [24], [26], [29]. Here we will use a semi-Lagrangian approximation based on a discrete time dynamic programming

---

[2]We can also consider the case where the objectives are switched, and the control wants to drive the system towards the target, by making minor modifications to the material presented here.

[3]A function $f : X \to Y$ between two measurable spaces (X, Σ_X) and (Y, Σ_Y) is said to be measurable if the preimage of a measurable set in Y is a measurable set in X, that is: $\forall V \in \Sigma_Y, f^{-1}(V) \in \Sigma_X$, with Σ_X, Σ_Y σ-algebras on X, Y.

[4] For any nonempty set $\mathcal{M} \subset \mathbb{R}^m$, the signed distance function $s_\mathcal{M} : \mathbb{R}^m \to \mathbb{R}$ is defined as $\inf_{y \in \mathcal{M}} |z - y|$ for points outside of $\mathcal{M}$ and $-\inf_{y \in \mathbb{R}^m \setminus \mathcal{M}} |z - y|$ for points inside $\mathcal{M}$, where $|\cdot|$ denotes a norm on $\mathbb{R}^m$.

[5]In this context reward refers to $l(x)$, but in general it can be any real-valued function.

(DP) principle:

$$V_{\Delta t}^{k+1}(x) = \min\{l(x), \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} V_{\Delta t}^k(x + \Delta t f(x, u, d))\},$$

(8a)

$$V_{\Delta t}^0(x) = l(x),$$ (8b)

$$V_{\Delta t} = \lim_{k \to \infty} V_{\Delta t}^k,$$ (8c)

where $V_{\Delta t}(x)$ converges to $V(x)$ as the discrete time step $\Delta t \to 0$. With the semi-Lagrangian approximation, the value function is solved on a discrete grid. Representing the approximation in vectorized form, $\vec{V} \in \mathbb{R}^{n_G}$, the semi-Lagrangian approach yields

$$\vec{V}_i^0 = l(x_i),$$ (9a)

$$\vec{V}_i^{k+1} = \min\{l(x_i), \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} I[\vec{V}^k](x_i + \Delta t f(x_i, u, d))\},$$

(9b)

$$\vec{V} = \lim_{k \to \infty} \vec{V}^k,$$ (9c)

for $i = 1, ..., n_G$, where $\{x_i\}_{i=1}^{n_G}$ are the grid nodes, n_G is the number of grid nodes, $\vec{V}_i^k$ is the approximate value for $V(x_i, k\Delta t)$ and $I[\vec{A}] : \mathbb{R}^n \to \mathbb{R}$ represents an interpolation operator defining, for every point x, the polynomial reconstruction based on the values $\vec{A}$. Unless stated otherwise G is taken to be a regular equidistant array of points with mesh spacing Δx_j along the jth axis, $j = 1, ..., n$.[6] We use a multilinear interpolator for the interpolation scheme, thus the interpolation function $I[\vec{A}](\cdot)$ is given by a convex combination over the elements of $\vec{A}$,

$$I[\vec{A}](x) = \phi(x) \cdot \vec{A},$$ (10)

where $\phi(\cdot) : \mathbb{R}^n \to \mathbb{R}^{n_G}$, and $\forall x \in \mathcal{R}^n$ the elements of $\phi(x)$ are nonnegative and sum to 1.

For conciseness we introduce *control policies* $\pi(\cdot) : \mathbb{R}^n \to \mathcal{U}$ and *disturbance policies* $\rho(\cdot) : \mathbb{R}^n \to \mathcal{D}$, which map from state to control and state to disturbance, respectively. Note that the individual minimax games being played at each grid point can be collectively thought of as a game over policies. We also introduce the *backup operator* $B[\cdot] : \mathbb{R}^{n_G} \to \mathbb{R}^{n_G}$, which maps vectorized value functions onto themselves. Define the backup operator as

$$B[\vec{A}] := \min\{\vec{l}, \max_{\pi} \min_{\rho} \Phi_{\pi,\rho} \vec{A}\},$$ (11)

where $\vec{l} \in \mathbb{R}^{n_G}$ with $\vec{l}_i = l(x_i)$ for $i = 1, ..., n_G$, $\Phi_{\pi,\rho} \in \mathbb{R}^{n_G \times n_G}$ is a policy-dependent stochastic matrix[7], and row i of $\Phi_{\pi,\rho}$ is $\phi(x_i + \Delta t f(x, \pi(x_i), \rho(x_i)))$. For the one player setting this matrix becomes Φ_π. We can now express (9) more concisely as

$$\vec{V}^0 = \vec{l},$$ (12a)

$$\vec{V}_i^{k+1} = B[V^k],$$ (12b)

$$\vec{V} = \lim_{k \to \infty} \vec{V}^k.$$ (12c)

[6]In the most general case the control and disturbance sets are also discretized, $\mathcal{U} = \{u_i\}_{i=1}^{n_{\mathcal{U}}}$ and $\mathcal{D} = \{d_i\}_{i=1}^{n_{\mathcal{D}}}$, and the minimax game is approximated.

[7]$\Phi_{\pi,\rho}$ is effectively the probability transition matrix for a Markov Decision Process (MDP) over a finite state space given by the grid nodes $\{x_i\}_{i=1}^{n_G}$.

This recursive procedure (12) is referred to as *value iteration*, and here $\vec{V}$ is the *vectorized value function*, which is used to approximate the value function $V(x)$ as $I[\vec{V}](x)$.

The value iteration algorithm in (12) can be used to solve other optimal control problems, albeit with a different backup operator and initialization $\vec{V}^0$. As other optimal control problems are presented we redefine the backup operator and specify the initialization required for the value iteration procedure.

D. Contraction Mappings and Sum of Discounted Rewards

Value iteration converges to a unique solution, independent of the initialization, when the backup operator is a *contraction mapping*.

Definition 1: A mapping $M(\cdot) : \mathbb{R}^{n_G} \to \mathbb{R}^{n_G}$, is said to be a contraction mapping in the norm $||\cdot||$ over the space $\mathbb{R}^{n_G}$ if there exists a Lipschitz constant $0 \leqslant \kappa < 1$ such that for any $\vec{A}_1, \vec{A}_2 \in \mathbb{R}^{n_G}$, $||M(\vec{A}_1) - M(\vec{A}_2)|| \leqslant \kappa ||\vec{A}_1 - \vec{A}_2||$.

Any contraction mapping has a unique fixed point. The operator given by (11) is not a contraction mapping. To see this note that any vector $\alpha \vec{1} \in \mathbb{R}^{n_G}$ is a fixed point for $\alpha < -L$.

The SDR problem does yield a contraction mapping, and we explore it briefly for the one player case. The objective is to maximize the integral (sum) of exponentially discounted rewards. In an abuse of notation, the associated value function $V(x)$ for this problem is given by

$$V(x) := \sup_{\boldsymbol{u} \in \mathbb{U}} \int_0^\infty r\left(\xi_x^{\boldsymbol{u}}(t)\right) \exp(-\lambda t) dt, \quad \lambda > 0,$$ (13)

where $r(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is a state-dependent reward function, and λ is a *discount rate*.

It is known that this value function is the solution to the time-independent Hamilton-Jacobi equation [4],

$$\lambda V(x) = \max_{u \in \mathcal{U}} \frac{\partial V}{\partial x}(x) f(x, u) + g(x).$$ (14)

Using the same semi-Lagrangian scheme as in the MR setting, (14) can be approximated as

$$V_{\Delta t}(x) = \max_{u \in \mathcal{U}} \gamma V_{\Delta t}(x + \Delta t \cdot f(x, u)) + \Delta t \cdot r(x),$$ (15)

where $\gamma = \exp(-\lambda \Delta t)$ is the *discount factor*.

The approximation is solved for using value iteration with the following backup operator:

$$B[\vec{A}] := \vec{r} + \max_{\pi} \gamma \Phi_\pi \vec{A},$$ (16)

where $\vec{r}_i = \Delta t \cdot r(x_i)$. The initialization of the value iteration procedure is arbitrary $\vec{V}^0 \in \mathbb{R}^{n_G}$ because the backup operator in (16) is a contraction mapping in the infinity norm, $||\cdot||_\infty$ [8].

Discounting reduces the impact of future rewards on the outcome of the trajectory, and ultimately yields a contraction mapping. In fact, the discount factor γ is the Lipschitz constant of the contraction mapping.

III. MINIMUM OF DISCOUNTED REWARDS

Inspired by the SDR setting, we now present the MDR optimal control problem. Moving forward any mention of value function $V(x)$ refers to the MR setting, and is defined by (4), (5), and (7).

A. What to discount?

A natural proposal for the outcome of the MDR problem would be

$$\inf_{t \geqslant 0} l\big(\xi_x^{\boldsymbol{u},\boldsymbol{d}}(t)\big) \exp(-\lambda t). \tag{17}$$

However, there is an issue with defining this outcome. Recall that discounting makes rewards contribute less to the outcome the further they occur in the future. Since we take an infimum, the discounted reward should become more positive the further it occurs in the future making it less likely to be selected by the infimum. This only works if the reward is nonpositive everywhere, which is not the case for l, since it is a clipped signed distance. This is easily fixed with the following outcome for the MDR problem

$$\mathcal{Z}\big(x, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)\big) := L + \inf_{t \geqslant 0}(l\big(\xi_x^{\boldsymbol{u},\boldsymbol{d}}(t)\big) - L) \exp(-\lambda t). \tag{18}$$

The quantity in the infimum is always nonpositive since L upperbounds $l(x)$ by construction. Note that if $\lambda = 0$, i.e. no discounting, then we have the minimum reward outcome (4).

We define the MDR value function as

$$Z(x) := \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}} \mathcal{Z}\big(x, \boldsymbol{u}(\cdot), \beta[\boldsymbol{u}](\cdot)\big). \tag{19}$$

For convenience we define two functions $U(x) := Z(x) - L$ and $h(x) := l(x) - L$. We will show that $U(x)$ is the viscosity solution to a particular time-independent HJ equation

$$0 = \min\left\{ h(x) - U(x), \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} \frac{\partial U}{\partial x}(x) f(x, u, d) - \lambda U(x)\right\}. \tag{20}$$

We begin by presenting some Lemmas to facilitate the proof.

Lemma 1: The function $U(x)$ is well defined.

Proof: Define the sequence $\{U(x, k)\}_k$, where

$$U(x, k) = \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}} \inf_{t \in [0, k\Delta t]} (h\big(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t)\big)) \exp(-\lambda t), \tag{21}$$

and $\Delta t > 0$. The sequence is nonincreasing, since $U(x, k+1) \leqslant U(x, k)$, and is lower bounded by $-2L$, so it converges. Clearly in the limit this sequence also equals $U(x)$. ∎

Lemma 2: Dynamic programming principle. For $\delta > 0$,

$$U(x) = \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}_\delta} \big[\min\{ \inf_{t \in [0, \delta]} h\big(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t)\big) \exp(-\lambda t),$$
$$\exp(-\lambda \delta) U(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta))\}\big], \tag{22}$$

where $\mathbb{U}_\delta$ consists of measurable functions on the interval $[0, \delta]$.

Proof: Splitting the time interval of the infimum in (18) into $[0, \delta]$ and $t > \delta$, $U(x)$ can be expressed as

$$U(x) = \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}} \big[\min\{ \inf_{t \in [0, \delta]} h\big(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t)\big) \exp(-\lambda t),$$
$$\inf_{t > \delta} h\big(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t)\big) \exp(-\lambda t)\}\big]. \tag{23}$$

Due to time-invariance of the dynamics, if we define $s = t - \delta$, $u_\delta(s) = u(t + \delta)$ and $y = \xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta)$,

$$U(x) = \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}} \big[\min\{ \inf_{t \in [0, \delta]} h\big(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t)\big) \exp(-\lambda t),$$
$$\exp(-\lambda \delta) \inf_{s > 0} h\big(\xi_y^{\boldsymbol{u}_\delta,\beta[\boldsymbol{u}_\delta]}(s)\big)) \exp(-\lambda s)\}\big]. \tag{24}$$

The game over the second interval can be optimized independently of the first interval once $y = \xi_x^{\boldsymbol{u},\boldsymbol{d}}(\delta)$ is specified thus it can be replaced with $\exp(-\lambda \delta) U(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta))$. Furthermore $\mathbb{U}$ is replaced with $\mathbb{U}_\delta$ since the game is only played explicitly on $[0, \delta]$. ∎

Before presenting the proof of the viscosity solution, it will be necessary to introduce the concepts of *viscosity subsolution* and *viscosity supersolution*. To simplify notation, we first define the *Hamiltonian* $H : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$,

$$H(x, p) = \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} f(x, u, d) \cdot p. \tag{25}$$

Definition 2: A function ϕ (in this case U) on $\mathbb{R}^n$ is a viscosity subsolution of (20), if for all $\psi \in C^1(\mathbb{R}^n)$ and x_0 such that $\phi(x_0) = \psi(x_0)$ and x_0 attains a local maximum on $\phi - \psi$, then

$$\min\left\{ h(x_0) - \psi(x_0), H(x_0, \frac{\partial \psi}{\partial x}) - \lambda \psi(x_0) \right\} \geqslant 0. \tag{26}$$

Definition 3: A function ϕ (in this case U) on $\mathbb{R}^n$ is a supersolution of (20), if for all $\psi \in C^1(\mathbb{R}^n)$ and x_0 such that $\phi(x_0) = \psi(x_0)$ and x_0 attains a local minimum on $\phi - \psi$, then

$$\min\left\{ h(x_0) - \psi(x_0), H(x_0, \frac{\partial \psi}{\partial x}) - \lambda \psi(x_0) \right\} \leqslant 0. \tag{27}$$

Now we present the major theoretical result for this section.

Theorem 1: The function $U(x)$ is the unique viscosity solution to the time-independent HJ equation given by (20).

Proof: The structure of the proof follows the classical approach in [16], analogously to [18], and draws from viscosity solution theory. We start by assuming that U is not a viscosity solution and then derive a contradiction to Lemma 2.

A continuous function is a viscosity solution if it is both a *subsolution* and *supersolution*. Note that U is uniformly continuous due to the continuity assumptions on f and l. We first show U is a subsolution of (20).

From the local maximum condition in Definition 2 and continuity of system trajectories, there exists a sufficiently small $\delta > 0$, such that for $\tau \in [0, \delta]$

$$U(\xi_{x_0}^{\boldsymbol{u},\boldsymbol{d}}(\tau)) \leqslant \psi(\xi_{x_0}^{\boldsymbol{u},\boldsymbol{d}}(\tau))$$

for all $u(\cdot) \in \mathbb{U}$ and $d(\cdot) \in \mathbb{D}$.

For sake of contradiction, assume (26) is false, then one of the following must be true

$$h(x_0) = \psi(x_0) - \epsilon_1 \tag{28a}$$
$$H(x_0, \frac{\partial \psi}{\partial x}) - \lambda \psi(x_0) = -\epsilon_2, \tag{28b}$$

for some $\epsilon_1, \epsilon_2 > 0$. If (28a) is true, then

$$h(\xi_{x_0}^{\boldsymbol{u},\boldsymbol{d}}(\tau)) \exp(-\lambda \tau) \leqslant \psi(x_0) - \frac{\epsilon_1}{2} = U(x_0) - \frac{\epsilon_1}{2} \tag{29}$$

Incorporating this into the dynamic programming principle (Lemma 2), we have

$$U(x_0) \leqslant \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}_\delta} \left\{ \inf_{t \in [0,\delta]} h\big(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t)\big)) \exp(-\lambda t) \right\}$$
$$\leqslant U(x_0) - \frac{\epsilon_1}{2}, \tag{30}$$

which is a contradiction since $\epsilon_1 > 0$. Similarly, if (28b), then for a small enough $\delta > 0$ and some nonanticipative strategy $\beta[\cdot]$,

$$H(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau), \frac{\partial \psi}{\partial x}) - \lambda \psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau)) \leqslant \frac{\epsilon_2}{2} \tag{31}$$

for all $\tau \in [0,\delta]$ and all inputs $\boldsymbol{u}(\cdot) \in \mathbb{U}$. Due to the $\max$ in (25), for $\tau \in [0,\delta]$

$$f(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau), \boldsymbol{u}(\tau), \beta[\boldsymbol{u}](\tau)) \cdot \frac{\partial \psi}{\partial x} - \lambda \psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau)) \leqslant$$
$$H(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau), \frac{\partial \psi}{\partial x}) - \lambda \psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau)). \tag{32}$$

Combining the two previous inequalities and integrating over the interval $[0,\delta]$ we have

$$\exp(-\lambda\delta)\psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta)) - \psi(x_0) \leqslant \frac{\epsilon_2}{2}\delta \tag{33}$$

Recalling that $U(x_0) = \psi(x_0)$

$$\exp(-\lambda\delta)U(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta)) \leqslant \frac{\epsilon_2}{2}\delta + U(x_0) \tag{34}$$

Incorporating this into the dynamic programming principle, we have

$$U(x_0) \leqslant \exp(-\lambda\delta)U(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta)) \leqslant \frac{\epsilon_2}{2}\delta + U(x_0), \tag{35}$$

which is a contradiction, thus we conclude that U is a subsolution.

Next we show that U is a supersolution. From the local minimum condition in Definition 3 and continuity of system trajectories, there exists a sufficiently small $\delta > 0$, such that for $\tau \in [0,\delta]$

$$U(\xi_{x_0}^{\boldsymbol{u},\boldsymbol{d}}(\tau)) \geqslant \psi(\xi_{x_0}^{\boldsymbol{u},\boldsymbol{d}}(\tau))$$

for all $u(\cdot) \in \mathbb{U}$ and $d(\cdot) \in \mathbb{D}$.

If we suppose (27) is false, then both of the following must hold

$$h(x_0) = \psi(x_0) + \epsilon_1 \tag{36a}$$
$$H(x_0, \frac{\partial \psi}{\partial x}) - \lambda \psi(x_0) = \epsilon_2, \tag{36b}$$

for some small $\epsilon_1, \epsilon_2 > 0$. If (36a) is true, then

$$h(\xi_{x_0}^{\boldsymbol{u},\boldsymbol{d}}(\tau)) \exp(-\lambda\tau) \geqslant \psi(x_0) + \frac{\epsilon_1}{2}$$
$$= U(x_0) + \frac{\epsilon_1}{2} \tag{37}$$

Similarly, if (28b), then for small enough $\delta > 0$ and some input $\boldsymbol{u}(\cdot) \in \mathbb{U}$

$$H(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau), \frac{\partial \psi}{\partial x}) - \lambda \psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau)) \geqslant \frac{\epsilon_2}{2} \tag{38}$$

for all $\tau \in [0,\delta]$ and all nonanticipative strategies $\beta[\cdot]$.

Due to the $\min$ in (25), for $\tau \in [0,\delta]$

$$f(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau), \boldsymbol{u}(\tau), \beta[\boldsymbol{u}](\tau)) \cdot \frac{\partial \psi}{\partial x} - \lambda \psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau)) \geqslant$$
$$H(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau), \frac{\partial \psi}{\partial x}) - \lambda \psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\tau)). \tag{39}$$

Combining the two previous inequalities and integrating over the interval $[0,\delta]$ we have

$$\exp(-\lambda\delta)\psi(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta)) - \psi(x_0) \geqslant \frac{\epsilon_2}{2}\delta \tag{40}$$

Recalling that $U(x_0) = \psi(x_0)$

$$\exp(-\lambda\delta)U(\xi_{x_0}^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta)) \geqslant \frac{\epsilon_2}{2}\delta + U(x_0). \tag{41}$$

Incorporating this into the dynamic programming principle, we have

$$U(x) = \inf_{\beta[\boldsymbol{u}](\cdot) \in \mathcal{B}} \sup_{\boldsymbol{u} \in \mathbb{U}_\delta} \big[\min\{ \inf_{t \in [0,\delta]} h(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(t))) \exp(-\lambda t),$$
$$\exp(-\lambda\delta)U(\xi_x^{\boldsymbol{u},\beta[\boldsymbol{u}]}(\delta))\}\big] \geqslant U(x) + \min\{\frac{\epsilon_1}{2}, \frac{\epsilon_2}{2}\delta\}, \tag{42}$$

which is a contradiction, thus U is also a supersolution.

Since we have shown that U is both a viscosity subsolution and viscosity supersolution of the HJ equation, this completes the proof that U is a viscosity solution of (20). Uniqueness follows from the classical comparison and uniqueness theorems for viscosity solutions (see Theorem 4.2 in [7]). ∎

B. Computing the Discounted Value Function

The discrete approximation of (20) is given by

$$U_{\Delta t}(x) = \min \left\{ h(x), \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} \gamma U_{\Delta t}(x + \Delta t f(x,u,d)) \right\}, \tag{43}$$

which can be solved on a grid G via value iteration

$$\vec{U}^0 \in \mathbb{R}^{n_G}, \tag{44a}$$
$$\vec{U}^{k+1} = B[U^k], \tag{44b}$$
$$\vec{U} = \lim_{k \to \infty} \vec{U}^k, \tag{44c}$$

with the backup operator defined as

$$B[\vec{A}] := \min \left\{ \vec{h}, \max_\pi \min_\rho \gamma \Phi_{\pi,\rho} \vec{A} \right\}, \tag{45}$$

where $\vec{h}_i = h(x_i)$. The MDR value function $Z(x)$ is then approximated by $I[\vec{U}](x) + L$, where again $I[\vec{U}](\cdot)$ is the interpolation operator. We now prove that (45) is a contraction.

Lemma 3: For any two functions $q, g : A \times B \to \mathbb{R}$,

$$|\max_a \min_b q(a,b) - \max_a \min_b g(a,b)| \leqslant \max_a \max_b |q(a,b) - g(a,b)|. \tag{46}$$

Proof: Define the minimax optimizers for q as the pair (a_q, b_q), and minimax optimizers of g as the pair (a_g, b_g). Without loss of generality we assume that $q(a_q, b_q) \geqslant g(a_g, b_g)$. We then have the following inequalities:

$$|\max_a \min_b q(a,b) - \max_a \min_b g(a,b)| \leqslant |q(a_q, b_q) - \min_b g(a_q, b)|$$
$$\leqslant |q(a_q, b_{gg}) - g(a_q, b_{gg})| \leqslant \max_a \max_b |q(a,b) - g(a,b)|,$$

with $b_{gg} := \arg \min_{b} g(a_q, b)$. ∎

Theorem 2: The operator given by (45) is a contraction mapping in the infinity norm $|| \cdot ||_\infty$ on the space $\mathbb{R}^{n_G}$.

Proof: Defining $B[\cdot]$ as in (45), take $A_1, A_2 \in \mathbb{R}^{n_G}$:

$$||B[\vec{A}_1] - B[\vec{A}_2]||_\infty =$$

$$|| \min \left\{ \vec{h}, \max_\pi \min_\rho \gamma \Phi_{\pi,\rho} \vec{A}_1 \right\} - \min \left\{ \vec{h}, \max_\pi \min_\rho \gamma \Phi_{\pi,\rho} \vec{A}_2 \right\} ||_\infty.$$

Leveraging the identity $\min\{a,b\} = \frac{1}{2}((a+b) - |a-b|)$ and using the shorthand $\Pi[\vec{A}] = \max_\pi \min_\rho \gamma \Phi_{\pi,\rho} \vec{A}$, the above is equal to

$$\frac{1}{2} ||(\Pi[\vec{A}_1] - \Pi[\vec{A}_2]) - (|\Pi[\vec{A}_1] - \vec{h}| - |\Pi[\vec{A}_2] - \vec{h}|)||_\infty,$$

which by the triangle inequality, is upper bounded by

$$\frac{1}{2} ||(\Pi[\vec{A}_1] - \Pi[\vec{A}_2])||_\infty + \frac{1}{2} ||(|\Pi[\vec{A}_1] - \vec{h}| - |\Pi[\vec{A}_2] - \vec{h}|)||_\infty.$$

Given the inequality $|a - b| > |(|a| - |b|)|$, this has upper bound

$$||(\Pi[\vec{A}_1] - \Pi[\vec{A}_2])||_\infty =$$

$$||\max_\pi \min_\rho \gamma \Phi_{\pi,\rho} \vec{A}_1 - \max_\pi \min_\rho \gamma \Phi_{\pi,\rho} \vec{A}_2||_\infty.$$

Finally from Lemma 3, the last upper bound is

$$\max_\pi \max_\rho ||\gamma \Phi_{\pi,\rho} (\vec{A}_1 - \vec{A}_2)||_\infty \leqslant \gamma ||\vec{A}_1 - \vec{A}_2||_\infty,$$

where the last inequality comes from the fact that $\Phi_{\pi,\rho}$ is a stochastic matrix for all policies, thus $||\Phi_{\pi,\rho}||_\infty = 1$. ∎

C. Under- and Over-Approximating the Reachable Set

With MDR formulation there is no particular level curve of the value function that characterizes the reachable set. However, it is possible to find level curves that correspond to over and under approximations of the reachable set.

We have the inequality $Z(x) \geqslant V(x)$ because the terms being discounted in the outcome are nonpositive. It immediately follows that

$$\{x \mid V_\lambda(x) \leqslant 0\} \subseteq \mathcal{R}(\mathcal{T}), \tag{47}$$

For an over-approximation we first need to characterize the error between $Z(x)$ and $V(x)$. The difference between the two functions can be bounded. Define $\tau(x)$ as the time when the minimum distance to the target is achieved for a trajectory starting at state x under the optimal control and disturbance signals. Then we have the following bound

$$Z(x) - V(x) \leqslant (L - l(\xi_x^{\boldsymbol{u},\boldsymbol{d}}(\tau(x))))(1 - \exp(-\lambda\tau(x))). \tag{48}$$

Noting that $V(x) = l(\xi_x^{\boldsymbol{u},\boldsymbol{d}}(\tau(x)))$, we get the resulting inequality

$$Z(x) - V(x) \exp(-\lambda\tau(x)) \leqslant L(1 - \exp(-\lambda\tau(x))), \tag{49}$$

Furthermore, outside the reachable set $V(x) > 0$ leading to

$$Z(x) - V(x) \leqslant L(1 - \exp(-\lambda\tau(x))) \quad \forall x \notin \mathcal{R}(\mathcal{T}). \tag{50}$$

Assuming an upper bound $\bar{\tau} \geqslant \tau(x)$, we have the following over-approximation for the reachable set

$$\mathcal{R}(\mathcal{T}) \subseteq \{x \mid Z(x) \leqslant L(1 - \exp(-\lambda\bar{\tau}))\}. \tag{51}$$

It is clear from (49) that the tightness of the approximations can be tuned via the discount rate λ.

IV. IMPROVING CONVERGENCE

In this section we present methods that may yield much faster convergence than value iteration. These approaches have been extensively applied to the SDR setting, and we now apply them to the MDR problem.

A. Policy Iteration

In the one player setting (control only), if the backup operator is a contraction mapping the solution can also be obtained via *policy iteration*. First define the *policy backup operator* $B^\pi[\cdot] : \mathbb{R}^{n_G} \to \mathbb{R}^{n_G}$,

$$B^\pi[\vec{A}] = \min \left\{ \vec{h}, \gamma \Phi_\pi \vec{A} \right\}. \tag{52}$$

This operator is a contraction mapping. The policy iteration algorithm generates the sequence $\{\vec{U}^{\pi^k}\}_k$ according to

$$\vec{U}^{\pi^k} = B^{\pi^k}[\vec{U}^{\pi^k}], \tag{53a}$$

$$\pi^{k+1} = \arg \max_\pi B^\pi[\vec{U}^{\pi^k}], \tag{53b}$$

$$\vec{U} = \lim_{k \to \infty} \vec{U}^{\pi^k}. \tag{53c}$$

Note that $\vec{U}^{\pi^k}$, the fixed point of (53a), is obtained through value iteration with the policy backup. Finding the fixed point of the policy backup is computationally less intensive than the other backup operators presented thus far, since no optimization is performed over policies. The policy iteration algorithm only optimizes over policies when switching policies.

In practice policy iteration is typically recommended over value iteration because policies can converge faster than values resulting in faster convergence of the algorithm [28]. A more detailed analysis of policy iteration (as it pertains to SDR) can be found in [9], [21], [27].

We now prove that policy iteration converges for the MDR problem.

Proposition 1: Assuming a finite control set $\mathcal{U} = \{u_i\}_{i=1}^{n_\mathcal{U}}$, the policy iteration algorithm converges to the vectorized value function obtained from (44) without the disturbance.

Proof: It's sufficient to show that sequence $\{\vec{U}^{\pi^k}\}_k$ is nondecreasing, i.e. $\vec{U}^{\pi^{k+1}} \geqslant \vec{U}^{\pi^k} \, \forall k$. Since the number of policies is finite the nondecreasing criterion implies that the sequence of vectors will converge. Also note that $\max_\pi B^\pi[\cdot] = B[\cdot]$ as defined in (44) without the disturbance, so the sequence converges to the vectorized value function.

Consider two sequences $\vec{X}^{i+1} = \min\{\vec{U}^{\pi^k}, \gamma \Phi_{\pi^{k+1}} \vec{X}^i\}$ and $\vec{Y}^{i+1} = \min\{\vec{h}, \gamma \Phi_{\pi^{k+1}} \vec{Y}^i\}$, with $\vec{X}^0 = \vec{Y}^0 = \vec{U}^{\pi^k}$. Since $\vec{h} \geqslant \vec{U}^{\pi^k}$, by (53a), we have $\vec{Y}^i \geqslant \vec{X}^i, \forall i \geqslant 0$. Next, we note that $\vec{X}^1 = \min\{\vec{U}^{\pi^k}, \gamma \Phi_{\pi^{k+1}} \vec{U}^{\pi^k}\} = \min\{\vec{h}, \gamma \Phi_{\pi^k} \vec{U}^{\pi^k}, \gamma \Phi_{\pi^{k+1}} \vec{U}^{\pi^k}\}$. Furthermore, the third term in the $\min\{\}$ is greater than the second, so

$\vec{X}^1 = \min\{\vec{h}, \gamma\Phi_{\pi^k}\vec{U}^{\pi^k}\} = \vec{U}^{\pi^k}$, thus $\vec{X}^i = \vec{U}^{\pi^k}, \forall i \geqslant 0$. Lastly, $\lim_{i\to\infty}\vec{Y}^i = \vec{U}^{\pi^{k+1}}$, which is the fixed-point of the contraction mapping that generates the sequence. Bringing everything together we have $\vec{U}^{\pi^{k+1}} = \lim_{i\to\infty}\vec{Y}^i \geqslant \lim_{i\to\infty}\vec{X}^i = \vec{U}^{\pi^k}$. ∎

B. Multigrid Approach

The accuracy of the approximation scheme depends on the fineness of the discretization. Finer grids have lower approximation error, but at the cost of increased computational effort. For a desired level of accuracy the number of grid points (and thus computation) necessary grows exponentially with the state space dimension, which is the well-known *curse of dimensionality*.

One possible way to manage this trade-off between computation and accuracy is a multigrid approach. The idea is to first solve for the approximation on a coarse grid (e.g. grid spacing $2\Delta x_j$) and then use the final solution to initialize either value iteration or policy iteration on a finer grid. The procedure can also be stacked, i.e. given m grids of increasing fineness we can produce approximations of increasing accuracy by using each as an initialization for the subsequent grid. This is only possible because contraction mappings allow great flexibility in the initialization, and yield faster convergence for good initializations. Due to the curse of dimensionality obtaining a good approximation is exponentially cheaper on the coarse grid.

Multigrid approaches have been applied extensively for SDR problems, where empirical and theoretical improvements have been shown [3], [11]. We compare multigrid approaches to value iteration in Section VI.

V. LEARNING REACHABLE SETS

When the system model is unknown or complex, the reachable set must be learned from data. There are two approaches for this: model-based RL and model-free RL. In this section we will focus on the model-based approach, and conclude with a brief discussion on the model-free approach and its connection to RL. For ease of presentation we consider the one player case.

A. Model-based

In the model-based approach the model is assumed to be parameterized by a parameter vector μ. The data is first used to fit the parameters, and then the value function is computed given the model. As more data is collected the process can be repeated. Here we are intentionally vague about the data and the fitting process, and we focus our attention on how to obtain the value function given the fitted model.

The data collection and fitting produce a sequence of parameters $\{\mu_k\}_k$, which in turn corresponds to a sequence of models $\{f_{\mu_k}\}_k$ and vectorized value functions $\{\vec{V}_{\mu_k}\}_k$ for the MR setting and $\{\vec{U}_{\mu_k}\}_k$ for the MDR setting. With the MR formulation the value iteration algorithm must be initialized with $\vec{l}$ every time a new value function is computed. However, with the MDR formulation $\vec{U}_{\mu_k}$ can be used as the initialization

when computing $\vec{U}_{\mu_{k+1}}$. Assuming regularity in the dynamics (with respect to μ), if the parameters only deviate slightly between iterations then $\vec{U}_{\mu_k}$ should be a good approximation of $\vec{U}_{\mu_{k+1}}$, resulting in faster convergence. If this is not the case then $\vec{l}$ can be used as the default initialization. Furthermore, the following classical result on contraction mappings can provide insight on selecting the initialization:

Proposition 2: If $M(\cdot) : \mathbb{R}^{n_G} \to \mathbb{R}^{n_G}$ is a contraction mapping in the norm $||\cdot||$ over the space $\mathbb{R}^{n_G}$ with Lipschitz constant $0 \leqslant \kappa < 1$ and fixed-point $\vec{A}^*$, then for any $\vec{A} \in \mathbb{R}^{n_G}$, $||\vec{A}^* - \vec{A}|| \leqslant \frac{1}{1-\kappa}||M(\vec{A}) - \vec{A}||$.

Given Proposition 2, when computing $\vec{U}_{\mu_{k+1}}$ an upper bound can be computed on its distance to $\vec{U}_{\mu_{k+1}}$ and $\vec{l}$ by applying the contraction mapping to each. The initialization can then be selected to minimize this upper bound. In the worst case only one additional backup operation is performed compared to the default case.

B. Model-free

Another approach to handling an unknown model, is to compute the value function directly from the data. This is the approach taken in many RL algorithms that attempt to approximate the value function for SDR problems with unknown models.

Temporal difference (TD) learning is at the heart of many of these methods, which includes TD-lambda[30], Q-learning[31], Deep Q Networks[25], and actor-critic methods[23]. The key idea is to represent the value function with a parametric function approximator[8], define a loss function on the parameters that is minimized when the value function is the fixed-point of the SDR backup operator, and to update the parameters by performing stochastic gradient descent on the loss function with samples from a real system or simulator.

For ease of presentation consider an autonomous system $f(x)$ (which might be due to a fixed policy), and a sequence of state transitions obtained from the system $\{(x_i, x_i^+)\}$, where $x^+ = \xi_x(\Delta t)$ and Δt is the time step. If the value function is approximated by $V_\theta(x)$ with parameters θ, then the loss function and update rule for TD are given by

$$\mathcal{L}(\theta) = \frac{1}{2}\big(V_\theta(x) - (r(x) + \gamma V_\theta(x^+))\big)^2,$$
$$\theta^{i+1} \leftarrow \theta^i + \alpha\nabla_\theta V_{\theta^i}(x_i)\big(r(x_i) + \gamma V_{\theta^i}(x_i^+) - V_{\theta^i}(x_i)\big),$$
$$(54)$$

where $\alpha \in [0,1]$ is the learning rate and θ^i is the parameters at iteration i. A similar idea can be used in the MDR setting. Taking the approximation $U_\theta(x)$ with parameter θ, the loss function and update rule for the MDR setting would be

$$\mathcal{L}(\theta) = \frac{1}{2}\big(U_\theta(x) - \min\{h(x), \gamma U_\theta(x^+)\}\big)^2,$$
$$\theta^{i+1} \leftarrow \theta^i + \alpha\nabla_\theta U_{\theta^i}(x_i)\big(\min\{h(x_i), \gamma U_{\theta^i}(x_i^+)\} - U_{\theta^i}(x_i)\big).$$
$$(55)$$

A similar modification can be made to the other TD-based algorithms, in particular those that take into account control

---

[8]A simple function approximator would be interpolation on a grid where the parameters are the grid node values.

actions. We leave the investigation of these ideas for future work.

VI. EXPERIMENTS

This section uses two benchmark models, double integrator and pursuit-evasion game, to exemplify the numerical properties of the MDR formulation. The double integrator model will be used to display the over-/under-approximation of the reachable set, as well as to compare policy iteration with value iteration. Both of the benchmarks will be used to demonstrate the advantages of multigridding, and initializing value iteration with pre-computed solutions to similar problems.

Unless stated otherwise all algorithms are initialized with $\vec{h} = \vec{l} - L$, and are considered converged when the distance (in the infinity norm) between consecutive iterates falls below $\epsilon = .001$. All experiments were run on a 2016 MacBook Pro with Core i7 processor and 16GB RAM.

A. Double Integrator

The doube integrator consists of two states (x_1, x_2) , and control $u \in [-u_{max}, u_{max}]$ with dynamics,

$$\dot{x_1} = x_2$$
$$\dot{x_2} = u \tag{56}$$

The state space is discretized into a 161×161 grid on the domain $[-1, 5] \times [-5, 5]$, and $u_{max} = 2$.

The task is to keep the state trajectory inside the box $\mathcal{K} = [0, 4] \times [-3, 3]$, thus the target is taken to be its complement $\mathcal{T} = \mathcal{K}^C$. For ease of exposition we define the *safe set* $\Omega(\mathcal{T}) := \mathcal{R}(\mathcal{T})^C$.

We first show, in Fig. 1 that different level curves of Z over approximates (bold line) and under approximates (dotted lines) the analytic safe set (shown in black) for two values of $\lambda = 0.1, 0.2$, with $\bar{\tau} = 2$. As expected smaller values of λ, yield tighter approximations.

To verify the convergence properties of the MDR formulation we initialize value iteration with the zero vector $\vec{0} \in \mathbb{R}^{N_G}$ with $\lambda = 0.1$. The error (in the infinity norm) between the converged solution and the one visualized in Fig. 1 is 0.000299, suggesting convergence to the same fixed point. Under the MR setting value iteration fails to converge with this particular initialization.

We now compare value iteration and policy iteration with increasing number of discrete actions in Table I. In the table we see that the runtime of value iteration increases linearly with the increase in the number of actions, and policy iteration scales much better. In our particular implementation, the overall runtime favors value iteration, but it is important to note that the majority of the time in policy iteration is spent constructing Φ_{π_u}, which is denoted by $T_{\Phi_{\pi_u}}$.[9] Excluding this cost, policy iteration becomes more attractive.

Next, we look at a multigrid approach versus value iteration. For the multigrid approach we also need to run value iteration on a coarse grid, which we construct to have half the resolution

[9]The data structure used to represent the interpolation is very efficient for sparse matrix multiplication, but is not ideal for indexing, which is necessary to create Φ_{π_u}, and results in a relatively large $T_{\Phi_{\pi_u}}$.
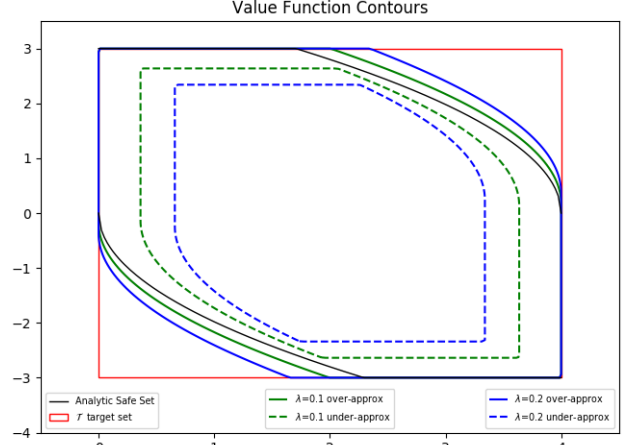

Fig. 1: The analytic safe set and target set $\mathcal{T}$ are shown in black (interior) and red (exterior), respectively. The over and under approximated Z are shown in bold green and dotted green for $\lambda = 0.1$; and bold blue and dotted blue line for $\lambda = 0.2$ (all interior).

TABLE I: Value Iteration vs Policy Iteration

# actions	VI	Policy Iteration	
	T_{total}	T_{total}	$T_{total} - T_{\Phi_{\pi_u}}$
2	1.468	78.197	0.102
50	8.753	302.456	1.007
250	36.973	308.565	4.318
500	65.305	326.280	9.760

per dimension of the nominal grid, e.g. if the nominal grid has 41^2 nodes then the coarse grid has 21^2 nodes. The results are shown in Table II. We first run value iteration with the standard initialization on both the coarse and fine grid. This produces the values in columns two and three. We then run value iteration on the fine grid initialized with the coarse solution (CS), which makes up column four. Column five (multigrid) is obtained by adding columns two and four. From the table it is clear that the multigrid approach outperforms value iteration, especially as the number of nodes increases.

Lastly, denoting the current model as the nominal model M_n, we construct two different models: a heavy model M_h with $u_{max} = 1.0$, and a light model M_l with $u_{max} = 4.0$. This can be interpreted as two systems that have different control authorities due to their different masses. The M_n MDR value function will be less than that of model M_l, but greater than that of M_h. We compute the value functions for M_l and M_h both initialized with the default initialization, and with the solution for model M_n, which we refer to as a *warm start* (WS). This experiment is motivated by Section V-A, where we discussed the computation of reachable sets

TABLE II: Double Integrator: Value iteration (VI) with Multigrid

# nodes	Coarse grid	Fine grid	Fine grid-CS	Multigrid
40^2	0.012	0.025	0.019	0.031
80^2	0.019	0.116	0.041	0.060
160^2	0.153	1.110	0.084	0.237

TABLE III: Double Integrator: Value Iteration (VI) with Warm Start (WS)

# nodes	M_n	M_l	M_l-WS	M_h	M_h-WS
40^2	0.029	0.044	0.041	0.022	0.014
80^2	0.205	.364	0.273	0.097	0.095
160^2	1.444	2.727	2.304	1.257	1.116

TABLE IV: Pursuit Evasion: Value iteration (VI) with Multigrid

# nodes	Coarse grid	Fine grid	Fine grid-WS	Multigrid
40^3	2.068	29.684	24.320	26.388
80^3	33.166	352.099	317.444	350.610

TABLE V: Pursuit Evasion: Value iteration (VI) with Warm Start (WS)

# nodes	M_n	M_e	M_e-WS	M_p	M_p-WS
40^3	35.043	32.242	21.483	26.319	23.366
80^3	439.751	416.965	308.821	300.847	296.568

as the system model changes due to new observations from the system. Here we are not concerned with how the model estimates are obtained, but rather how to produce the reachable set for the latest model estimate as quickly as possible. Within this context M_n can be viewed as an old model, and M_l and M_h can be thought of as two possible new models that were inferred from observations. The results for the experiment are shown in Table III. In both cases (M_l and M_h) we see that leveraging the solution for M_n improves the convergence time.

B. Pursuit-Evasion Game

We now consider the pursuit-evasion game described in [24]. In the game player I (the control) tries to avoid being captured by player II (the disturbance) on a two dimensional plane. Each player is modeled as a simple kinematic point object with planar position and heading, fixed linear velocity and controllable angular velocity. Taking player I to be at the origin the states (x_1, x_2, x_3) are the relative position and heading of player II and the dynamics are

$$\dot{x_1} = -v_u + v_d \cos x_3 + u x_2$$
$$\dot{x_2} = v_d \sin x_3 - u x_1 \quad (57)$$
$$\dot{x_3} = d - u$$

The state space is over the domain $[-6, 20] \times [-10, 10] \times [0, 2\pi[$ with $\mathcal{U} = [-u_{max}, u_{max}]$ and $\mathcal{D} = [-d_{max}, d_{max}]$.

Player I is considered captured when the relative distance (in position) between both players is less than $R > 0$, thus the target set is given by

$$\mathcal{T} = \{x | x_1^2 + x_2^2 < R^2\} \quad (58)$$

We first compute the value functions for the MR and MDR on a $41 \times 41 \times 41$ grid, and setting the model parameters to $v_u = v_d = 5$, $u_{max} = d_{max} = 1$, and $R = 5$. This will be referred to as the nominal model M_n. A visualization of the zero sub-level set for both $V(x)$ and $Z(x)$ for $\lambda = 0.001$ is shown in Fig. 2.

In the first experiment we compare a multigrid approach to value iteration. The results are shown in Table IV. The experiment and table follows the same structure used for the double integrator model. Similar to the double integrator model, the multigrid approach outperforms value iteration for the pursuit-evasion game.

We now construct two other models by tweaking M_n: setting $u_{max} = 1.5$, which gives the evader an advantage, we get model M_e, and setting $d_{max} = 1.5$, which gives the pursuer an advantage, we get model M_p. In the final experiment we look at the impact of initializing value iteration

with a solution from a similar model. Just like in the previous benchmark example, this experiment is motivated by Section V-A, where now M_e and M_p represent two possible models inferred from the system observations. In this context we have just "learned" that the evader/pursuer is more maneuverable (M_e/M_p). We compute both value functions with and without setting the initialization to the solution for M_n. Again, we refer to this initialization as a warm start. The results are shown in Table V.

VII. CONCLUSIONS AND FUTURE WORK

We have presented a novel minimum discounted reward HJ formulation for approximating reachable sets. The main advantage of this new formulation over previous work is that the solution can be obtained as the unique fixed point to a contraction mapping. We also showed how other solutions like policy iteration, and multigrid approaches can be used to yield faster convergence.

The benefits listed so far, are within the context of the traditional control paradigm, where we have or assume a fixed model of the system under consideration. However, as learning and data-driven approaches become more powerful and pervasive, perhaps the greatest contribution of this work is that it can lie somewhere in between traditional control and model-free reinforcement learning. The approach is certainly model-based, but because of its agnosticism to initialization it also has the flexibility to incorporate data and build towards solutions iteratively. We foreshadowed how this can be done with temporal difference learning, and in the future we plan on exploring how RL algorithms can be used with this formulation to approximate reachable sets for systems with unknown models or that are high-dimensional.

REFERENCES

[1] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, "Reachability-based safe learning with gaussian processes," *Proceedings of 53rd IEEE Conference on Decision and Control*, 2014.

[2] A. K. Akametalu and C. J. Tomlin, "Temporal-difference learning for online reachability analysis," *Control Conference (ECC), 2015 European*, IEEE, 2015.

[3] A. Alla, M. Falcone, and D. Kalise, "An efficient policy iteration algorithm for dynamic programming equations," *SIAM Journal on Scientific Computing*, 2015.

[4] M. Bardi and I. Capuzzo-Dolcetta, *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Springer Science & Business Media, 2008.
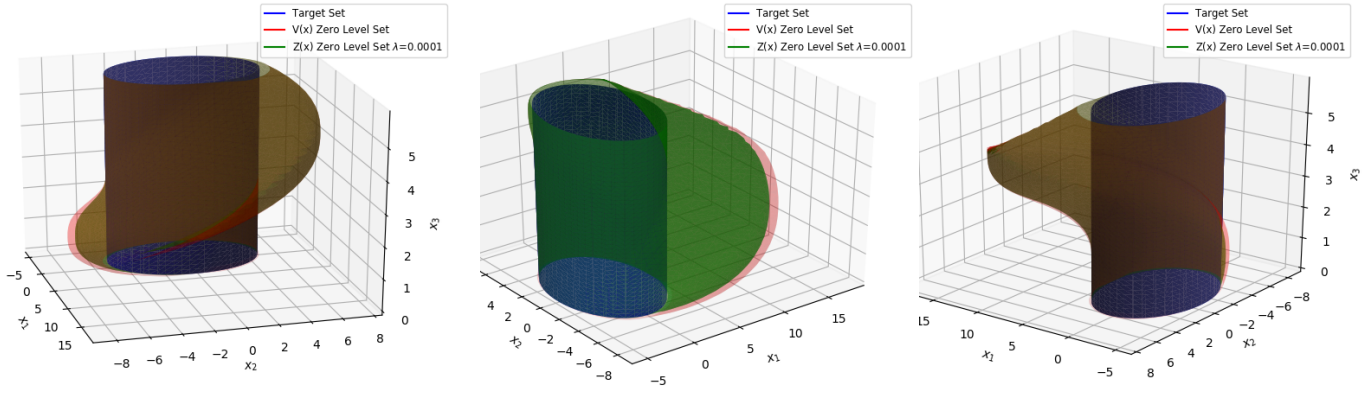
Fig. 2: The target set $\mathcal{T}$ (blue cylinder), zero sub-level sets of V (red) and Z (green) shown from three different perspectives. The discount rate for Z is $\lambda = 0.01$. Note that the zero sub-level set of Z is a subset of the zero sub-level set of V.

[5] M. Bardi, M. Falcone, and P. Soravia, "Numerical methods for pursuit-evasion games via viscosity solutions," *Stochastic and differential games*, Springer, 1999.

[6] E. Barron, "Differential games with maximum cost," *Nonlinear analysis: Theory, methods & applications*, 1990.

[7] E. Barron and H. Ishii, "The bellman equation for minimizing the maximum cost," *Nonlinear Analysis: Theory, Methods & Applications*, 1989.

[8] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995.

[9] O. Bokanowski, S. Maroso, and H. Zidani, "Some convergence results for howard's algorithm," *SIAM Journal on Numerical Analysis*, 2009.

[10] M. Chen, Q. Hu, C. Mackin, J. F. Fisac, and C. J. Tomlin, "Safe platooning of unmanned aerial vehicles via reachability," *Conference on Decision and Control (CDC)*, 2015.

[11] C Chow and J. N. Tsitsiklis, "An optimal one-way multigrid algorithm for discrete-time stochastic control," *IEEE transactions on automatic control*, 1991.

[12] E. A. Coddington and N. Levinson, *Theory of ordinary differential equations*. Tata McGraw-Hill, 1955.

[13] J. Ding, E. Li, H. Huang, and C. J. Tomlin, "Reachability-based synthesis of feedback policies for motion planning under bounded disturbances," *International Conference on Robotics and Automation (ICRA)*, 2011.

[14] J. Ding, C. J. Tomlin, L. R. Hook, and J. Fuller, "Initial designs for an automatic forced landing system for safer inclusion of small unmanned air vehicles into the national airspace," *Proceedings of the 2016 IEEE/AIAA 35th Digital Avionics Systems Conference*, IEEE, 2016.

[15] B. Djeridane and J. Lygeros, "Neural approximation of pde solutions: An application to reachability computations," *Decision and Control, 2006 45th IEEE Conference on*, IEEE, 2006.

[16] L. C. Evans and P. E. Souganidis, "Differential games and representation formulas for solutions of hamilton-jacobi-isaacs equations," *Indiana University mathematics journal*, 1984.

[17] M. Falcone and R. Ferretti, "Discrete time high-order schemes for viscosity solutions of hamilton-jacobi-bellman equations," *Numerische Mathematik*, 1994.

[18] J. F. Fisac, M. Chen, C. J. Tomlin, and S. S. Sastry, "Reach-avoid problems with time-varying dynamics, targets and constraints," *Proceedings of the 18th international conference on hybrid systems: Computation and control*, ACM, 2015.

[19] J. H. Gillula and C. J. Tomlin, "Guaranteed safe online learning via reachability: tracking a ground target using a quadrotor," *International Conference on Robotics and Automation (ICRA)*, 2012.

[20] G. M. Hoffmann and C. J. Tomlin, "Decentralized cooperative collision avoidance for acceleration constrained vehicles," *2008 47th IEEE Conference on Decision and Control*, 2008.

[21] R. A. Howard, *Dynamic programming and Markov processes*. Wiley for The Massachusetts Institute of Technology, 1964.

[22] H. Huang, J. Ding, W. Zhang, and C. J. Tomlin, "A differential game approach to planning in adversarial scenarios: a case study on capture-the-flag," *International Conference on Robotics and Automation (ICRA)*, 2011.

[23] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," *Advances in neural information processing systems*, 2000.

[24] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," 2005.

[25] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," *International Conference on Machine Learning*, 2016.

[26] S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*. Springer Science & Business Media, 2003.

[27] M. L. Puterman and S. L. Brumelle, "On the convergence of policy iteration in stationary dynamic programming," *Mathematics of Operations Research*, 1979.

[28] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, *Artificial intelligence: A modern approach*. Prentice hall Upper Saddle River, 2003.

[29] J. Sethian, "A fast marching level set method for monotonically advancing fronts.," *Proceedings of the National Academy of Sciences of the United States of America*, 1996.

[30] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine learning*, 1988.

[31] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, 1992.