Responses to reviewer 11

Corrections to your comments are highlighted in color ■. In the revised version, we have made the following changes to the symbols and terminologies used in the manuscript.

Original version	**Revised version**
Controller,disturbance	Player I, Player II
$a \in \mathcal{A}$, $d \in \mathcal{D}$	$a \in \mathcal{A}$, $b \in \mathcal{B}$
$\mathscr{A}$, $\mathscr{D}$	$\mathscr{A}_0$, $\mathscr{B}_0$
Given time horizon $\bar{T}$	T or $T_1, ..., T_M$
Given admissible cost $\bar{J}$	J or $J_1, ..., J_M$
$\phi_{t_0}^{t_1}(., s_0, a, d)$	$\phi_{s,t}^{a,b}(.)$
$\min\limits_{a(.)} \max\limits_{d(.)}\ t_f$	$\sup\limits_{\beta(.)} \inf\limits_{a(.)}\ t_f$ (Emphasize that Player II plays nonanticipative strategies)
$J_{t_0}^{t_1}(s_0, a, d)$	$\mathcal{J}_{t_0}^{t_1}(s, a, b)$
$\min\limits_{a(.)} \max\limits_{d(.)}\ J_{t_0}^{t_1}(s_0, a, d)$	$\sup\limits_{\beta(.)} \inf\limits_{a(.)}\ \mathcal{J}_{t_0}^{t_1}(s, a, \beta[a])$
Given final time t_f	$\bar{T}$ (The symbol of the free final time t_f remains unchanged)
$\widehat{W}_{t_f}^c(.)$	$W_{\bar{T}}^c(., 0)$, $\left(W_{\bar{T}}^c(s, \tau) = \sup\limits_{\beta(.)} \inf\limits_{a(.)}\ \mathcal{J}_{\tau}^{\bar{T}}(s, a, \beta[a])\right)$

1. Overall, the contributions are not very clear. It is hard to understand the comparison of this method in terms of the stability issues and the advantages with the level-set method. In particular, in which way the proposed recursion is more stable? In which way the discretization approach is consistent?

Response: The potential instability of the level set method, which involves viscosity solutions of the HJ PDE, arises from two aspects: unsuitable time step size and unsuitable dissipation function. The time step is determined by the CFL condition and the suitable dissipation function of the ith dimension is determined by:

$$\alpha_i = \max \left| \frac{\partial}{\partial p_i} \left[\min_a \max_b p^{\mathrm{T}} f(s, a, b) \right] \right| \tag{1}$$

The documentation for the Level Set Toolbox suggest that computing the dissipation function is a tricky task [2]. In addition, the dissipation function are related to the spatial derivatives of $V(.,.)$, i.e. $\frac{\partial V}{\partial s}$. However, in our approach, the value function $W_{\bar{T}}^c(.,.)$ is often discontinuous, in which case the left or right derivative cannot be defined and the difference between the grid points does not converge. Take a one-dimensional problem as an example:

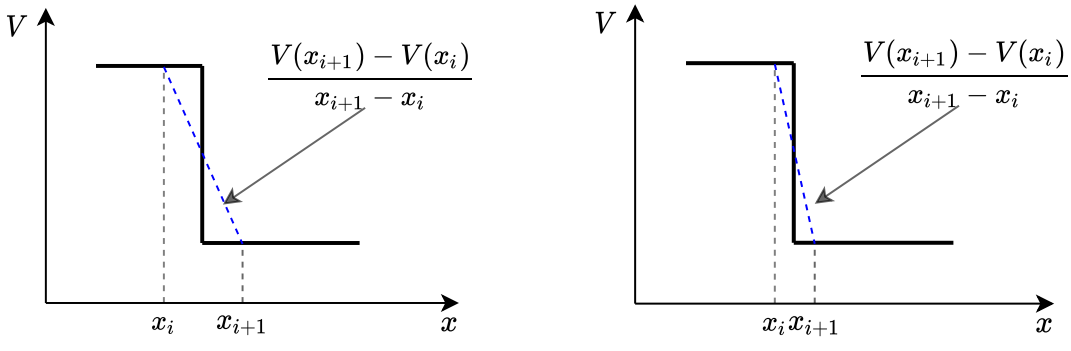


Figure 1:

As the grid size decreases, the difference between grid points near discontinuous points becomes larger. It is because of this discontinuity that we chose a method based on recursion and interpolation rather

than a viscosity solution. Recursion and interpolation-based method does not require the computation of spatial derivatives. Take a two-dimensional problem as an example, given different $a \in \mathcal{A}$ and $b \in \mathcal{B}$, the system will transfer to different states after one time step, see the following figure.

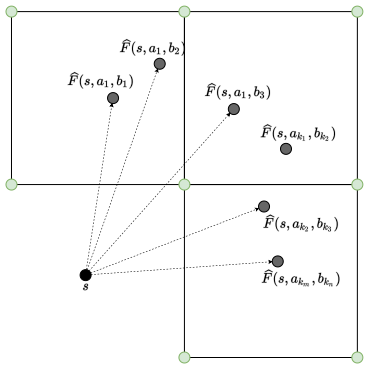


Figure 2:

To compute $W^c_{\bar{T}}(s_0, k\Delta t)$, two terms need to be computed, one is the cost consumed during the transition $\widehat{C}(s_0, a, b)$ and the other is the value of the function at the transferred state $W^c_{\bar{T}}(\widehat{F}(s_0, a, b), (k+1)\Delta t)$. $W^c_{\bar{T}}(\widehat{F}(s_0, a, b), (k+1)\Delta t)$ can be obtained by the bilinear interpolation of the four nearest grid points to $\widehat{F}(s_0, a, d)$. Therefore, both parts do not involve the spatial derivatives or the numerical stability problems common in the process of numerically solving PDEs.

Therefore, to some extent, it can be argued that the use of recursion and interpolation is more a choice of last resort than an advantage of the proposed method. This is caused by the discontinuity of the value function. But this value function also brings great benefits, one of which is the small storage space consumption. In the level set method, to compute the BRT of time horizon T, one needs to solve the following HJ PDE:

$$\begin{cases} \dfrac{\partial V}{\partial t}(s,t) + \min\left[0, \min\limits_{a\in\mathcal{A}} \max\limits_{b\in\mathcal{B}} \dfrac{\partial V}{\partial s}(s,t) f(s,a,b)\right] = 0 \\ \text{s.t.} V(s,T) = l(s) \end{cases} \tag{2}$$

Then the BRT is:

$$R_K(T) = \{s_0 \in \mathbb{R}^n | V(s_0, 0) \leq 0\} \tag{3}$$

This means that to save $R_K(T)$, one needs to save value function $V(., 0)$. In this case, the storage space consumed is proportional to $\prod_{i=1}^n N_i$, where N_i is the number of grid points in the ith dimension of the Cartesian grid. For $T_1, ..., T_M \in [0, \infty)$, the representations of the BRTs of these time horizons are:

$$\begin{aligned} &R_K(T_1) = \{s_0 \in \mathbb{R}^n | V(s_0, T - T_1) \leq 0\} \\ &... \\ &R_K(T_M) = \{s_0 \in \mathbb{R}^n | V(s_0, T - T_M) \leq 0\} \end{aligned} \tag{4}$$

Since the value functions $V(., T - T_1), ..., V(., T - T_M)$ are each different, the storage space consumption required to save these BRTs is proportional to $M \prod_{i=1}^n N_i$, which is M times as much as the original. In the proposed method, only $W^c_{\bar{T}}(., 0)$ ($\bar{T} > \max(T_1, ..., T_M)$) needs to be saved to save $R_K(T_1), ..., R_K(T_M)$. These BRTs are represented as:

$$\begin{aligned} &R_K(T_1) = \left\{s_0 \in \mathbb{R}^n | W^c_{\bar{T}}(s_0, 0) < T_1\right\} \\ &... \\ &R_K(T_M) = \left\{s_0 \in \mathbb{R}^n | W^c_{\bar{T}}(s_0, 0) < T_M\right\} \end{aligned} \tag{5}$$

The storage space consumed does not vary with M and is always proportional to $\prod_{i=1}^n N_i$, which is one part in M of that of the level set method. We visually display the superiority of this storage approach in the first example of the revised manuscript.

In the revision, we replaced the time-discrete form from forward Euler method to fourth-order Runge-Kutta method. The results show that the error decreases as the grid size decreases, and the convergence rate is comparable to that of the level set method. see Fig. 4 in the revision.

In summary, in the revised version we have recapitulated the main contributions of this paper, in particular the second innovation, which puts more emphasis on the advantages of the constructed value function itself rather than on the method of computing this value function.

2. Furthermore, it seems that given the cost-limited formulation, the conversion of the problem to a reachability problem with an extended state-space is straightforward. In particular, we can add a state that accounts for the running cost. Then, the problem of not exceeding the cost level while reaching a target set can be formulated as a reachability for the extended state (there the target set is the product of the original one and that of [0, c], with c being the upper bound on the cost). Here are few additional points that can be clarifed:

Response: This is theoretically possible, but adding one dimension to the state space means that the computation time and memory consumption increases by a factor of tens or even hundreds (depending on the number of grids in this added dimension)

3. In Section 2.1, the level set approach for backward reachability is reviewed. However, the instability issues referred to in the introduction are not discussed. It'd be helpful to clarify what is the dissipation function in the formulation of (4), what is the challenge in computing it as mentioned in the introduction and why the instabilities arise. This helps motivate the authors' proposed approach.

Response: In the level set method, the solution of the HJ PDE is not differentiable everywhere. Therefore, a viscosity solution is introduced. The spatial derivative is replaced by the left and right derivatives [2,3], take the direction along x axis as an example:

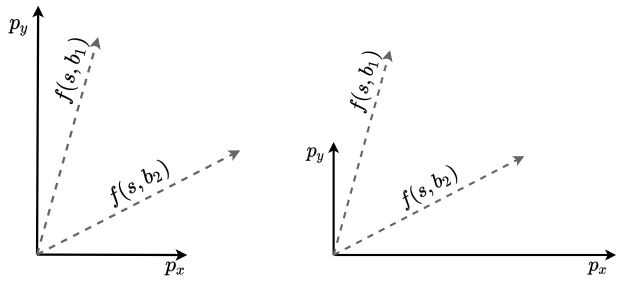
$$\min_{a\in\mathcal{A}}\max_{b\in\mathcal{B}} p_x f_x(s,a,b) = \min_{a\in\mathcal{A}}\max_{b\in\mathcal{B}}\left[\frac{p_x^+ + p_x^-}{2} f_x(s,a,b)\right] - \alpha_x \frac{p_x^+ - p_x^-}{2} \tag{6}$$

where $p_x = \frac{\partial V}{\partial x}$, $f_x(s,a,b)$ is the component of $f(s,a,b)$ along x axis, p_x^- and p_x^+ are the left and right derivatives, respectively, α_x is the dissipation function, and

$$\alpha_x = \max\left|\frac{\partial}{\partial p_x}\left[\min_a \max_b p^{\mathrm{T}} f(s,a,b)\right]\right| \tag{7}$$

From the above equation, to compute α_x, it is necessary to traverse the set $\mathcal{A}$ and $\mathcal{B}$, as well as the entire computational domain, because the choice of a and b is related to p. For the sake of simplicity, consider the following equation:

$$\max_b p^{\mathrm{T}} f(s,b) \tag{8}$$



(a) (b)

Figure 3:

In the case shown in Fig. 3(a), $\arg \max_{b\in\{b_1,b_2\}} p^{\mathrm{T}} f(s,b) = b_1$, in the case shown in Fig. 3(b), $\arg \max_{b\in\{b_1,b_2\}} p^{\mathrm{T}} f(s,b) = b_2$. A too small α_x will lead to instability of the viscous solution and too large α_x will lead to computational errors. Therefore, overestimation of α_x is the preferable choice. If the system is of affine nonlinear form, i.e.,

$$f_x(s,a,d) = f_x^s(s) + g_1(s)a + g_2(s)b \tag{9}$$

α_x can be overestimated by [2]:

$$\alpha_x \leq |f_x^s(s)| + \max\left[|g_1(s)|\, a_{\max}, |g_1(s)|\, a_{\min}\right] + \max\left[|g_2(s)|\, b_{\max}, |g_2(s)|\, b_{\min}\right] \tag{10}$$

As previously mentioned, the method in this paper does not involve viscosity solutions, and there is no need to compute the dissipation function. Since the reason for not having to compute the dissipation function is that this paper uses a recursion-interpolation-based approach, and the use of recursion and interpolation cannot be considered as an advantage of the proposed method, the description of the dissipation function has been removed in the revised version. In the revised version, we put more emphasis on the properties of the value function itself, rather than on the method of computing the value function.

4. It is hard to understand what optimization problem (6) is saying. So, if there exists and input that satisfies the constraints, the cost is min max t_f and otherwise, infinity? It'd be good to clarify. Furthermore, as far as I am aware, this is not a standard optimal control problem. It'd be helpful to comment about this and its connection with other optimal control problems.

Response: Strictly speaking, this is a differential game. t_f is the final time and t_f is free. t_f is the first hitting time to the target set, Player I aims to minimize it and Player II amis to maximize it. This performance index is a special form of that in Eq. (12) of the revision, i.e.

$$c(s,a) \equiv 1 \Longrightarrow \int_0^{t_f} c(s(t),a(t))dt = t_f \tag{11}$$

See Fig. 1 of the revision. The description of the problem has been revised in the revision. For some states, there exists a $b(.) \in \mathscr{B}_0$ such that no matter what Player I does, the trajectories from these states never reach the target set. In the first example of the manuscript, let $s_0 = [-1,1]^{\mathrm{T}}$, if $b(t) \equiv -1$, no matter how the control input of Player I is chosen, the trajectory from s_0 can never reach the target set. For such a s_0, the value function $W(s_0) = \infty$ or $W^c(s_0) = \infty$. Because no matter how large the given time horizon T or admissible cost J is, the BRT or CBRT will not contain s_0, i.e.:

$$s_0 \notin \{s \in \mathbb{R}^n | W(s) \leq T\} \forall T \in [0,\infty) \text{ and } s_0 \notin \{s \in \mathbb{R}^n | W^c(s) \leq J\} \forall J \in [0,\infty) \tag{12}$$

5. It'd be helpful to explain why (20) holds and how this result connects with dynamic programming.

Response: We are very sorry that we forgot to give the definition of s_0', s_0' is the transferred state of the trajectory from s_0 after time τ, i.e. $s_0' = \phi_0^{t_f}(\tau, s_0, a, d)$, it depends on $a(.) \in \mathscr{A}$ and $d(.) \in \mathscr{D}$ in time interval $[0,\tau]$.

$a(.)$ and $d(.)$ are chosen to consider the sum of cumulative cost of over time interval $[0,\tau]$, and the cumulative cost over time interval $[\tau, t_f]$, i.e.

$$\begin{aligned} &\min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \hat{J}_0^{t_f}(s_0,a,d) = \min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \int_0^{t_f} \hat{c}\left(\hat{\phi}_0^{t_f}(t,s_0,a,d),a(t)\right) dt \\ &= \min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \left[\int_0^{\tau} \hat{c}\left(\hat{\phi}_0^{t_f}(t,s_0,a,d),a(t)\right) dt + \int_{\tau}^{t_f} \hat{c}\left(\hat{\phi}_0^{t_f}(t,s_0,a,d),a(t)\right) dt\right] \end{aligned} \tag{13}$$

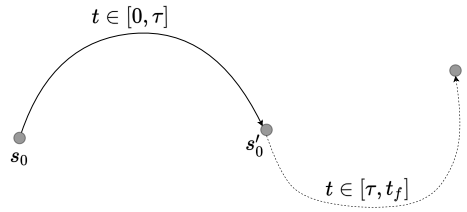


Figure 4:

The integration interval of the definite integral $\int_{\tau}^{t_f} \hat{c}\left(\hat{\phi}_0^{t_f}(t, s_0, a, d), a(t)\right) dt$ is $[\tau, t_f]$, therefore,

$$\int_{\tau}^{t_f} \hat{c}\left(\hat{\phi}_0^{t_f}(t, s_0, a, d), a(t)\right) dt = \int_{\tau}^{t_f} \hat{c}\left(\hat{\phi}_{\tau}^{t_f}(t, s'_0, a, d), a(t)\right) dt \tag{14}$$

Since the control input and disturbance on interval $[0, \tau]$ and those on interval $[\tau, t_f]$ are independent, together with the fact that the system is non-time-varying, the above equation can be rewritten as:

$$\int_{\tau}^{t_f} \hat{c}\left(\hat{\phi}_0^{t_f}(t, s_0, a, d), a(t)\right) dt = \int_{0}^{t_f-\tau} \hat{c}\left(\hat{\phi}_0^{t_f-\tau}(t, s'_0, a_{-\tau}, d_{-\tau}), a_{-\tau}(t)\right) dt \tag{15}$$

where $a_{-\tau}(t) = a(\tau + t)$, $d_{-\tau}(t) = d(\tau + t)$. Combining Eq. (13), Eq. (14) and Eq. (15), the following equation can be obtained:

$$\begin{aligned} &\min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \hat{J}_0^{t_f}(s_0, a, d) \\ &= \min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \left[\int_0^{\tau} \hat{c}\left(\hat{\phi}_0^{t_f}(t, s_0, a, d), a(t)\right) dt + \int_0^{t_f-\tau} \hat{c}\left(\hat{\phi}_0^{t_f-\tau}(t, s'_0, a_{-\tau}, d_{-\tau}), a_{-\tau}(t)\right) dt\right] \end{aligned} \tag{16}$$

Since the control input and disturbance on $[0, \tau]$ and those on interval $[\tau, t_f]$ are independent, the control inputs and disturbances on interval $[0, \tau]$ and interval $[\tau, t_f]$ can be considered as two independent pairs of variables. By replacing $a_{-\tau}(.)$ with $a'(.)$ and $d_{-\tau}(.)$ with $d'(.)$, the above equation becomes

$$\begin{aligned} \min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \hat{J}_0^{t_f}(s_0, a, d) = &\min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \left[\int_0^{\tau} \hat{c}\left(\hat{\phi}_0^{t_f}(t, s_0, a, d), a(t)\right) dt \right. \\ &\left. + \min_{a'(.)\in\mathscr{A}} \min_{d'(.)\in\mathscr{D}} \int_0^{t_f-\tau} \hat{c}\left(\hat{\phi}_0^{t_f-\tau}(t, s'_0, a_{-\tau}, d_{-\tau}), a_{-\tau}(t)\right) dt\right] \\ = &\min_{a(.)\in\mathscr{A}} \min_{d(.)\in\mathscr{D}} \left[\int_0^{\tau} \hat{c}\left(\hat{\phi}_0^{t_f}(t, s_0, a, d), a(t)\right) dt + \widehat{W}^c_{t_f-\tau}(s'_0)\right] \end{aligned} \tag{17}$$

In the revised version, we refer to Theorem 3.1 of Paper [1] to derive the recursive formula.

6. Furthermore, it is not clear why this min max problem is easier than the recursion involved in the level-set method. In particular, often level set methods also discretize the state-space and a similar recursion needs to be solved.

Response: The level set method involves viscosity solutions of the HJ PDE, therefore, to avoid numerical instability, it is necessary to consider the CFL condition as well as the dissipation function, which are not involved in our method. In addition, the HJ PDE in the level set method can be written in the following backward recursive form:

$$\begin{aligned} &V(s, T) = l(s) \\ &V(s, t - \Delta t) = \min\left[V(s, t), \min_{a\in\mathcal{A}} \max_{b\in\mathcal{B}} V\left(F(s, a, b), t\right)\right] \end{aligned} \tag{18}$$

where $s(t+\Delta t) = F(s(t), a(t), b(t))$ is the time discrete form of $\dot{s} = f(s, a, b)$. Our recursive form is:

$$\begin{aligned} &W_{\bar{T}}^c(s, \bar{T}) = 0 \\ &W_{\bar{T}}^c(s, t-\Delta t) = \min_{a\in\mathcal{A}} \max_{b\in\mathcal{B}} \left[\widehat{C}(s,a,b) + W_{\bar{T}}^c(\widehat{F}(s,a,b), t)\right] \end{aligned} \tag{19}$$

As can be seen, the two recursive forms are quite different. It is the value function generated by this recursive form that has the above-mentioned advantage in terms of storage space. Therefore, in the revised version, we put more emphasis on the advantages of the constructed value function itself, rather than on the method of computing this value function.

7. Are there any consistency results? In particular, as the number of discretization points increase, the solution converges to the optimal one with certain rate?

Response: In the revision, we replaced the time-discrete form from forward Euler method to fourth-order Runge-Kutta method. The results show that the error decreases as the grid size decreases, and the convergence rate is comparable to that of the level set method. see Fig. 4 in the revision.

8. In the first example, what is the considered cost? Also, please explain where the analytic solution is coming from.

Response: The first example computes the BRT. As described in Remark 1, CBRT degenerates to BRT when running cost $c(s, a) \equiv 1$. So in this example, the running cost function is $c(s, a) \equiv 1$.

The derivation of the analytic solution is as follows: After removing the control inputs of both players, the vector field of the dynamical system is shown below:

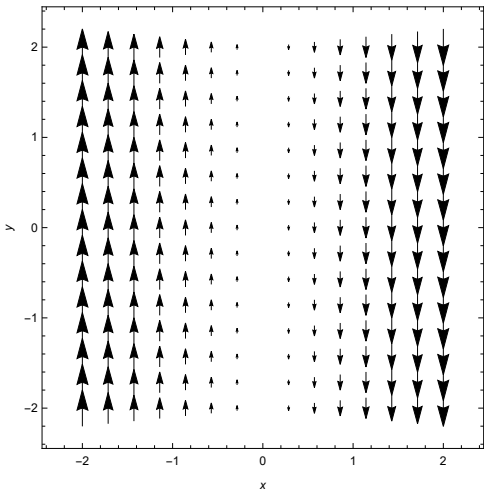


Figure 5: The vector field of the dynamical system

For a state $s_0 = [x_0, y_0]^{\mathrm{T}}$,
Case (1) $y_0 \geq 0$:

In order to drive the state to the target set as quickly as possible, the best policy for Player I is to drive the state down the y-axis. To avoid or delay the entry of states into the target set, the best policy for Player II is to drive the states to the left along the x-axis. Therefore, in this case, the optimal policies for controller and disturbance are:

$$y_0 > 0.5 \Longrightarrow a^*(s_0) = -1, b^*(s_0) = -1 \tag{20}$$

Under these policies, the variation of y with time t is:

$$y(t) = \frac{1}{2}t^2 - x_0 t - t + y_0 \tag{21}$$

This is a parabola in the (y,t) plane, and its axis of symmetry is $\tilde{t} = x_0 + 1$. If $[x_0, y_0]^{\mathrm{T}} \in R_K(1)$, then

$$y(1) \leq 0.5 \text{ or } y(\tilde{t}) \leq 0.5, 0 \leq \tilde{t} \leq 1 \tag{22}$$

Solving the above inequalities yields:

$$[x_0, y_0]^{\mathrm{T}} \in \left\{[x_0, y_0]^{\mathrm{T}} | 0 \leq y_0 \leq x_0 + 1\right\} \cup \left\{[x_0, y_0]^{\mathrm{T}} | 0 \leq y_0 \leq \frac{1}{2}x_0^2 + x_0 + 1, -1 \leq x_0 \leq 0\right\} \tag{23}$$

Case (2) $y_0 < 0$:

Similar to the derivation of Case (1), in this case, the optimal policies for controller and disturbance are:

$$y_0 > 0.5 \Longrightarrow a^*(s_0) = 1, b^*(s_0) = 1 \tag{24}$$

then

$$y(t) = -\frac{1}{2}t^2 - x_0 t + t + y_0 \tag{25}$$

The axis of symmetry is $\tilde{t} = -x_0 + 1$. Then

$$[x_0, y_0]^{\mathrm{T}} \in R_K(1) \Longrightarrow y(1) \geq -0.5 \text{ or } y(\tilde{t}) \geq -0.5, 0 \leq \tilde{t} \leq 1 \tag{26}$$

Solving the above inequalities yields:

$$[x_0, y_0]^{\mathrm{T}} \in \left\{[x_0, y_0]^{\mathrm{T}} | 0 \geq y_0 \geq x_0 - 1\right\} \cup \left\{[x_0, y_0]^{\mathrm{T}} | 0 \geq y_0 \geq -\frac{1}{2}x_0^2 + x_0 - 1, 0 \leq x_0 \leq 1\right\} \tag{27}$$

The target set K is also a subset of the BRT, therefore,

$$\begin{aligned} R_K(1) = K \cup \left\{[x_0, y_0]^{\mathrm{T}} | 0 \leq y_0 \leq x_0 + 1\right\} \cup \left\{[x_0, y_0]^{\mathrm{T}} | 0 \leq y_0 \leq \frac{1}{2}x_0^2 + x_0 + 1, -1 \leq x_0 \leq 0\right\} \cup \\ \left\{[x_0, y_0]^{\mathrm{T}} | 0 \geq y_0 \geq x_0 - 1\right\} \cup \left\{[x_0, y_0]^{\mathrm{T}} | 0 \geq y_0 \geq -\frac{1}{2}x_0^2 + x_0 - 1, 0 \leq x_0 \leq 1\right\} \end{aligned} \tag{28}$$

This analytic solution is also verified by the level set method.

9. In example 2, I understand it is hard to compare with level-set methods since those methods do not account for a cost function. Nevertheless, perhaps you can provide the comparison without inclusion of costs?

Response: In example 2, the running cost is

$$c(s, \delta_e) = 1 + \lambda_G \|G\|_2 = 1 + \frac{\lambda_G}{mg}\sqrt{|F_z|^2 + |F_x|^2} \tag{29}$$

Two cases is studied: $\lambda_G = 0$ and $\lambda_G = 0.25$ (see the sentence "We consider two cases:..."). When $\lambda_G = 0$, the running cost is $c(s, \delta_e) \equiv 1$. According to Remark 1, the CBRT becomes BRT. It is perhaps important to point out that the running cost function is not present in the level set method, but is present in our method (Our method can compute the BRT when setting the running cost function to a constant 1 instead of deleting it). The result of case $\lambda_G = 0$ is compared to that of the level set method.

References

[1] P. Souganidis L. Evans. Differential games and representation formulas for solutions of hamilton-jacobi-isaacs equations. *Indiana Univ. Math. J.*, 33:773–797, 1984.

[2] Ian M Mitchell. A toolbox of level set methods. *UBC Department of Computer Science Technical Report TR-2007-11*, 2007.

[3] Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153. Springer Science & Business Media, 2006.