A multilevel fast-marching method

Marianne Akian, Stéphane Gaubert, Shanqing Liu

A multilevel fast-marching method

Marianne Akian, Stéphane Gaubert * **Shanqing Liu** **

* *Inria and CMAP, École polytechnique, IP Paris, CNRS, 91128 Palaiseau Cedex, France (e-mail: Firstname.Name@inria.fr)*
** *CMAP, École polytechnique, IP Paris, CNRS, and Inria, 91128 Palaiseau Cedex, France (e-mail: Shanqing.Liu@polytechnique.edu)*

Abstract: We introduce a new numerical method to approximate the solutions of a class of static Hamilton-Jacobi-Bellman equations arising from minimum time optimal control problems. We rely on several grid approximations, and look for the optimal trajectories by using the coarse grid approximations to reduce the search space for the optimal trajectories in fine grids. This may be thought of as an infinite dimensional version, for PDE, of the "highway hierarchy" method which has been developed to solve discrete shortest path problems. We obtain, for each level, an approximate value function on a sub-domain of the state space. We show that the sequence obtained in this way does converge to the viscosity solution of the HJB equation. Moreover, the number of arithmetic operations that we need to obtain an error of $O(\varepsilon)$ is bounded by $\tilde{O}(1/\varepsilon^{\frac{2d}{1+\beta}})$, to be compared with $\tilde{O}(1/\varepsilon^{2d})$ for ordinary grid-based methods. Here $\beta \in (0,1]$ depends on the "stiffness" of the value function around optimal trajectories, and the notation $\tilde{O}$ ignores logarithmic factors. Under a regularity condition on the dynamics, we obtain a bound of $\tilde{O}(1/\varepsilon^{(1-\beta)d})$ operations, for $\beta < 1$, and this bound becomes $O(|\log\varepsilon|)$ for $\beta = 1$. This allowed us to solve HJB PDE of eikonal type up to dimension 7.

Keywords: Minimum Time, Eikonal equation, Fast Marching Method, Highway Hierarchies.

1. INTRODUCTION

We consider the minimal time optimal control problem, consisting in finding a trajectory minimizing the travel time between two points. The minimal time, together with optimal trajectories, can be obtained by solving a Hamilton-Jacobi-Bellman (HJB) partial differential equation (PDE) of eikonal type. Such an equation is obtained from the dynamic programming principle, and has to be thought in viscosity sense, see Fleming and Soner (2006).

One of the most famous methods to solve the eikonal PDE is the fast-marching method, see Sethian (1996). Though it is computationally efficient, it is still a grid-based method, and hence suffers from the "curse of dimensionality" of the dynamical programming approach. Our algorithm intends to bypass this difficulty by narrowing the search space around optimal trajectories, hence solving a series of PDE, in which the domain is becoming smaller and smaller with the discretize grid becoming finer and finer. This is inspired by the recent development of the "Highway Hierachies" algorithm, which applies to shortest path problems in discrete time and in discrete space, see Sanders and Schultes (2012).

The idea of our algorithm is as follows: instead of computing the optimal trajectories directly in the whole domain, we first find a subdomain which contains the optimal trajectories, then do the search in such a subdomain. This will be achieved by using coarse and fine grids discretization method. We first use a coarse grid to discretize the whole domain, and do a partial fast marching search in this coarse grid. Then, we find a subdomain which contains the true optimal trajectories, by using the approximate value function on the coarse grid, together with the error bound. Finally we use a finer grid to discretize the subdomain, and perform a fast marching search on this fine grid. We repeat this operation, considering different levels of finer and finer grids.

We show that using our algorithm, the final approximation error is as good as the one obtained by directly discretizing the whole domain with the finest grid. Moreover, the number of elementary operations and the size of the memory needed to get an error of ε are considerably reduced. Indeed, let $\kappa(M) := M \log M$, and recall that the fast marching method implemented in a d-dimensional grid with M points requires a number of arithmetic operations of order $\kappa(M)K_d$, in which the constant $K_d \in [2d, 2^d]$ depends on the stencil of the discretization used for the fast marching method. For our method, the number of arithmetic operations is in the order of $\kappa((\frac{1}{\varepsilon})^{\frac{2d}{1+\beta}})C^d$, where $C > 1$ and $\beta \in (0,1]$ are fixed constants (see Section 5), to be compared with $\kappa((\frac{1}{\varepsilon})^{2d})K_d$ for conventional grid-based methods. Moreover, under a regularity condition on the dynamics, our complexity bound reduces to an order of $\kappa((\frac{1}{\varepsilon})^{(1-\beta)d})C^d$ for $\beta < 1$, and $\kappa(d|\log\varepsilon|)$ for $\beta = 1$.

2. HAMILTON-JACOBI EQUATION FOR MINIMUM TIME PROBLEM

Let Ω be an open, bounded domain in $\mathbb{R}^n$. Let S_1 be the unit sphere in $\mathbb{R}^n$, i.e., $S_1 = \{x \in \mathbb{R}^n, \|x\| = 1\}$, where $\|\cdot\|$ denotes the Euclidean norm. Let $\mathcal{A} = \{\alpha : [0, +\infty] \to S_1 : \alpha(\cdot)$ is measurable$\}$ denote the set of controls. We denote

by f the speed function, and assume the following basic regularity assumption:

Assumption 2.1.

i. $f : \overline{\Omega} \times S_1 \to (0, +\infty)$ is continuous.
ii. There exists positive constants $L_f, \gamma, L_{f,\alpha}$ such that $|f(x,\alpha) - f(x',\alpha)| \leqslant L_f|x - x'|$, and $|f(x,\alpha) - f(x,\alpha')| \leqslant L_{f,\alpha}|\alpha - \alpha'|^\gamma, \forall x, x' \in \Omega, \alpha, \alpha' \in S_1$.

Our goal is to find the minimum time necessary to travel from the source point $x_{\text{src}} \in \Omega$ to the destination point $x_{\text{dst}} \in \Omega$, and the optimal trajectories, together with the optimal control α. We consider the controlled dynamical system:

$$\dot{x}(t) = f(x(t), \alpha(t))\alpha(t) \ . \tag{1}$$

We denote by $x_{\alpha,x_{\text{src}}}$, or simply x_α, the solution of (1) with $\alpha \in \mathcal{A}$, such that $x_\alpha(0) = x_{\text{src}}$ and $x_\alpha(s) \in \overline{\Omega}$ for all $0 \leqslant s \leqslant t$. We restrict the set of control trajectories so that the state x_α stays inside the domain $\overline{\Omega}$ forever, i.e., we consider the following set of admissible controls trajectories:

$$\mathcal{A}_{\Omega,x_{\text{src}}} := \{\alpha \in \mathcal{A} \mid x_{\alpha,x_{\text{src}}}(s) \in \overline{\Omega}, \forall s \geqslant 0\},$$

and we further assume $A_{\Omega,x_{\text{src}}} \neq \emptyset$. In other words, the structure of $A_{\Omega,x_{\text{src}}}$ is adapted to the state constraint $x_\alpha(s) \in \overline{\Omega}$. By doing so, the minimum time function can be defined as

$$T_{\text{s}\to}(x) = \inf_{\alpha \in \mathcal{A}_{\Omega,x_{\text{src}}}} \inf\{\tau \mid x_{\alpha,x_{\text{src}}}(\tau) = x\} \ .$$

Consider the Kruzkov change of variable:

$$v_{\text{s}\to}(x) = 1 - e^{-T_{\text{s}\to}(x)} \ .$$

Then, $v_{\text{s}\to}(x)$ is the viscosity solution of the following state constrained HJB equation:

$$\begin{cases} F(x, v_{\text{s}\to}(x), Dv_{\text{s}\to}(x)) = 0, \ x \in \Omega, \\ F(x, v_{\text{s}\to}(x), Dv_{\text{s}\to}(x)) \geqslant 0, \ x \in \partial\Omega, \\ v_{\text{s}\to}(x_{\text{src}}) = 0 \ . \end{cases} \tag{2}$$

where $F(x, r, p) = -\min_{\alpha \in S_1}\{-p \cdot f(x,\alpha)\alpha + 1 - r\}$.

3. THE OPTIMAL TRAJECTORIES OF THE CONTINUOUS SPACE PROBLEM

In this section we show how to reduce the state space Ω of the original minimum time problem, while preserving the optimal trajectories.

Definition 3.1. For every $x, y \in \Omega$, we denote by $\Gamma(x,y)$ the set of *geodesic points* from x to y, defined as

$$\Gamma(x,y) = \{x_{\alpha,x}(s) \mid s \in [0,\tau], \alpha \in \mathcal{A}_{\Omega,x}, x_{\alpha,x}(\tau) = y \text{ and } \int_0^\tau e^{-s}ds = \inf_{\tau \geqslant 0, \alpha \in \mathcal{A}_{\Omega,x},\ x_{\alpha,x}(\tau)=y} \{\int_0^\tau e^{-s}ds\}\}.$$

In other words, z is a geodesic point between x and y if an optimal trajectory from x to y passes through z.

Let us consider a new minimal time optimal control problem ending in x_{dst} with the same dynamics as (1), but starting at any $x \in \overline{\Omega}$. The associated minimal time function is

$$T_{\to\text{t}}(x) = \inf_{\alpha \in \mathcal{A}_{\Omega,x}} \inf\{\tau \mid x_{\alpha,x}(\tau) = x_{\text{dst}}\} \ .$$

By doing so, we have $T_{\text{s}\to}(x_{\text{dst}}) = T_{\to\text{t}}(x_{\text{src}})$, and we denote $\tau^* = T_{\text{s}\to}(x_{\text{dst}}) = T_{\to\text{t}}(x_{\text{src}})$. We then use the same change of variable technique to get $v_{\to\text{t}}(x) = 1 - e^{-T_{\to\text{t}}(x)}$.

Let us denote:

$$\mathcal{F}_v(x) := v_{\text{s}\to}(x) + v_{\to\text{t}}(x) - v_{\text{s}\to}(x)v_{\to\text{t}}(x) \ .$$

Let $\Gamma^* = \Gamma(x_{\text{src}}, x_{\text{dst}})$ denote the union of all optimal trajectories from x_{src} to x_{dst}. Then, by the dynamic programming principle, the following result holds :

Lemma 3.1. We have

$$v_{\text{s}\to}(x_{\text{dst}}) = v_{\to\text{t}}(x_{\text{src}}) = \inf_{y \in \Omega} \mathcal{F}_v(y) \ . \tag{3}$$

Moreover, if Γ^* is not empty, then for every $x \in \Gamma^*$ we have $\mathcal{F}_v(x) = v_{\text{s}\to}(x_{\text{dst}})$, that is x is optimal in (3). If there exists an optimal trajectory between any two points of Ω, then x is optimal in (3), that is $\mathcal{F}_v(x) = v_{\text{s}\to}(x_{\text{dst}})$ if and only if $x \in \Gamma^*$.

Let us now consider an open subdomain of Ω, $\mathcal{O}_\eta \subseteq \Omega$, determined by a parameter $\eta > 0$, and defined as follows:

$$\mathcal{O}_\eta = \{x \in \Omega \mid \mathcal{F}_v(x) < \inf_{y \in \Omega}\{\mathcal{F}_v(y) + \eta\}\} \ . \tag{4}$$

We intend to reduce the state space of our original optimal control problem from Ω to $\mathcal{O}_\eta$. More precisely, we consider a new optimal control problem with the same dynamics, but we restrict the controls so that the state x_α stays inside the domain $\mathcal{O}_\eta$, leading to the new set of controls:

$$\mathcal{A}_{\eta,x_{\text{src}}} := \{\alpha \in \mathcal{A} \mid x_{\alpha,x_{\text{src}}}(s) \in \overline{O_\eta}, \forall s \geqslant 0\}.$$

Let $v^\eta_{\text{s}\to}(x)$ denote the value function of the new problem, then $v^\eta_{\text{s}\to}(x)$ is a viscosity solution of the following HJB equation:

$$\begin{cases} F(x, v^\eta_{\text{s}\to}(x), Dv^\eta_{\text{s}\to}(x)) = 0, \ x \in O_\eta, \\ F(x, v^\eta_{\text{s}\to}(x), Dv^\eta_{\text{s}\to}(x)) \geqslant 0, \ x \in \partial O_\eta, \\ v^\eta_{\text{s}\to}(x_{\text{src}}) = 0 \ . \end{cases} \tag{5}$$

Then we have the following result:

Proposition 3.1. If Γ^* is not empty, then $\Gamma^* \subseteq \mathcal{O}_\eta$, and for all $x \in \Gamma^*$ we have $v_{\text{s}\to}(x) = v^\eta_{\text{s}\to}(x), v_{\to\text{t}}(x) = v^\eta_{\to\text{t}}(x)$.

The above results express properties of exact optimal trajectories. We will also consider approximate, $\delta-$optimal, trajectories:

Definition 3.2. For every $x, y \in \Omega$, for any $\delta > 0$, we denote $\Gamma_\delta(x,y)$ the set of $\delta-$geodesic points from x to y , defined as :

$$\Gamma_\delta(x,y) = \{x_{\alpha,x}(s) \mid s \in [0,\tau], \alpha \in \mathcal{A}_{\Omega,x}, x_{\alpha,x}(\tau) = y, \text{ and } \int_0^\tau e^{-s}ds \leqslant \inf_{\tau \geqslant 0, \alpha \in \mathcal{A}_{\Omega,x},\ x_{\alpha,x}(\tau)=y} \{\int_0^\tau e^{-s}ds\} + \delta\} \ .$$

Let $\Gamma^*_\delta = \Gamma_\delta(x_{\text{src}}, x_{\text{dst}})$ denote the set of all $\delta-$geodesic points from x_{src} to x_{dst}. Then we have the following results:

Lemma 3.2. For every $\eta > \delta > 0$, we have $\Gamma^*_\delta \subseteq \mathcal{O}_\eta$.

Lemma 3.3. For every $\eta > 0$, there exists $\delta' < \eta$ such that $\mathcal{O}_\eta \subseteq \Gamma^*_{\eta+\delta'}$.

The above two lemmas entail that the sets of $\delta-$geodesic points and $\mathcal{O}_\eta$ constitute two equivalent families of neighborhoods. Thanks to these properties, we establish the following result :

Theorem 3.1. For every $x \in \Gamma^*_\delta$, we have

$$v^\eta_{\text{s}\to}(x) = v_{\text{s}\to}(x), \quad v^\eta_{\to\text{t}}(x) = v_{\to\text{t}}(x) \ .$$

Thus, if we are only interested to find $v_{s\to}(x_{\text{dst}})$ and the optimal trajectories between x_{src} and x_{dst}, we only need to solve the reduced problem (5) in the subdomain $\mathcal{O}_\eta$. This is equivalent to solving the original problem in Ω as long as $\mathcal{O}_\eta$ contains the optimal trajectories of this problem.

4. THE MULTI-LEVEL FAST-MARCHING ALGORITHM

4.1 Motivation

The main idea of our algorithm is based on the property proposed above. Instead of computing the optimal trajectories directly, we first find an approximate subdomain of O_η by applying the fast marching search in a coarse grid, with relaxed accuracy and error requirements. Then, we further discretize O_η using a finer grid, and perform a search on this fine grid.

4.2 The Update Operator for The Fast-Marching Search

Based on Assumption 2.1, there exist constants $\underline{f}, \overline{f}$ such that: $0 < \underline{f} \leqslant f(x,\alpha) \leqslant \overline{f} < \infty$. We define $\Upsilon := \frac{\overline{f}}{\underline{f}} \geqslant 1$, and observe that this constant can be interpreted as a measure of anisotropy of the minimum time problem.

Let $X = \Omega \cap (h\mathbb{Z})^d$, we define $V_{s\to}$ as the approximation of the value function $v_{s\to}$ in X. Let $x \in X$, for any I adjacent nodes $x_1, x_2, ..., x_I$ in the grid X, we define $x_\rho = \sum_{i=1}^I \rho_i x_i$ for $\rho \in \Delta^I = \{\rho_i \geqslant 0, \sum_{i=1}^I \rho_i = 1\}$. Let us denote $d(\rho) = \|x_\rho - x\|$ and $\alpha_\rho = \frac{x_\rho - x}{\|x_\rho - x\|}$, which are the distance and the direction from x to x_ρ.

Let $\mathcal{I} = \{1, 2, ..., I\}$. Let $V_{s\to}(x; (x_i)_{i\in\mathcal{I}})$ denote the approximate value $V_{s\to}(x)$ of $v_{s\to}(x)$ depending on the nodes $(x_i)_{i\in\mathcal{I}}$. Then, applying a semi-Lagrangian discretization scheme, we have:

$$V_{s\to}(x; (x_i)_{i\in\mathcal{I}}) = \min_{\rho\in\Delta^I}\{(1 - \frac{d(\rho)}{f(x,\alpha_\rho)}) \sum_i (\rho_i V_{s\to}(x_i)) + \frac{d(\rho)}{f(x,\alpha_\rho)}\} .$$

Let us denote $N(x)$ the set of neighborhood nodes of x, defined as follows:

$$N(x) := \{x_j \in X \mid \exists x_k \in X, s.t. \ \exists \tilde{x} \in [x_j, x_k],\ \|\tilde{x} - x\| \leqslant \Upsilon h\},$$

where we use $[a, b]$ to denote the line segment between a and b. Then, the update operator for the value function in the fast-marching method is as follows:

$$\mathcal{U}(V_{s\to}(x)) := \min\{V_{s\to}(x), \min_{(x_i)_{i\in\mathcal{I}}\in N(x)} V_{s\to}(x; (x_i)_{i\in\mathcal{I}})\} .$$

4.3 The Algorithm

Two Level Fast Marching. We start by describing the special case of the algorithm with only two levels of grid. The algorithm consists of three main steps: Step 1. Discretize Ω using the grid X^H, and find a good $O_\eta^{H,I} \subseteq \Omega$. Step 2. Discretize $O_\eta^{H,I}$ using the fine grid O_η^h. Step 3. Doing a fast marching search in grid O_η^h. We give the details of the first two steps:

Step 1, discretization in coarse grid. We start with the full domain Ω. Let $X^H = \Omega \cap (H\mathbb{Z})^d$. Without loss of generality, we assume $x_{\text{src}}, x_{\text{dst}} \in X^H$.

By doing a fast marching search in X^H starting from x_{src} and x_{dst} respectively, we get the numerical approximation $V_{s\to}^H$ for $v_{s\to}$, and $V_{\to t}^H$ for $v_{\to t}$. We use the approximate value functions $V_{s\to}^H$ and $V_{\to t}^H$ to construct a subset of X^H, denoted by O_η^H:

$$O_\eta^H = \{x^H \in X^H \mid \mathcal{F}_{V^H}(x^H) \leqslant \min_{x^H\in X^H} \mathcal{F}_{V^H}(x^H) + \eta^H\}.$$

Then we construct a continuous analogue of the grid O_η^H, denoted by $O_\eta^{H,I}$, by defining the approximate value functions $V_{s\to}$ and $V_{\to t}$ on the whole domain using linear interpolation of the functions $V_{s\to}^H$ and $V_{\to t}^H$.

Step 2, discretization in fine grid-h. Let the finite set X^h denote the approximation of Ω with mesh step h. Without loss of generality we assume $x_{\text{src}}, x_{\text{dst}} \in X^h$. Let the finite set O_η^h be the approximation of $O_\eta^{H,I}$ with the mesh step h, given by:

$$O_\eta^h = \{x^h \in X^h \mid \exists x^H \in O_\eta^H : \|x^h - x^H\| \leqslant \max((H-h), h)\}.$$

Multi-Level Grids. The computation in the 2-level method above can be easily extended to the Multi-Level case. For each level l, we consider the optimal control problem in which the domain Ω is restricted to the fine-grid region $O_{\eta^{l-1}}^{H_{l-1},I}$ of the previous level $l-1$. Then, by applying the first two steps of the 2-level algorithm as above, we get a new domain, $O_{\eta^l}^{H_l,I}$, taken to be the new state space for the optimal control problem to be solved at the next level. At the end, we do a fast-marching search starting from x_{src} in the final fine grid.

4.4 Implementation

Algorithm 1 Two-Level Fast-Marching Method

Input: Mesh step of coarse and fine grids: H, h. The parameter η^H. The update operator for fast-marching method: $\mathcal{U}$. Start and end point: $x_{\text{src}}, x_{\text{dst}}$.
Output: Approximated value function: $V_{s\to}^h(x)$.

```
Do the fast-marching search staring from x_src and x_dst.
for Every node x^H ∈ X^H do
    if F_{V^H}(x) ⩽ min_{x^H∈X^H} F_{V^H}(x^H) + η^H then
        Set x^H as ACTIVE , store it's position.
    end if
end for
Begin with set FINE be empty.
for Every node x^H in the ACTIVE set do
    for Every x^h ∈ X^h : ‖x^h − x^H‖_∞ ⩽ max{H − h, h} do
        if x^h does not exist in set FINE then
            Add x^h in the set FINE, store it's position.
        end if
    end for
end for
Do fast-marching search starting from x_src in FINE.
```

In Algorithm 2, the selection of active nodes corresponds to Step 1 in the two-level algorithm, the selection of fine grid corresponds to Step 2 in the two-level algorithm.

Algorithm 2 Multi-Level Fast-Marching Method

Input: The parameter H_l, η_l, for $l \in \{1, 2, ..., N\}$. The update operator for fast-marching scheme: $\mathcal{U}$. Start and end point: $x_{\text{src}}, x_{\text{dst}}$.

Output: Approximated value function: $V^h_{\text{s}\to}(x)$.

Let X^{H_1} be the COARSE-GRID.
for $l = 1$ to $N - 1$ **do**
    Do the partial fast marching search starting from x_{src} and x_{dst}.
    Select the ACTIVE nodes from the COARSE-GRID.
    Select the set FINE nodes based on the ACTIVE nodes, as in Algorithm 1.
    Let the FINE be the new COARSE-GRID.
end for
Do the fast-marching search starting from x_{src} in FINE.

To implement efficiently these algorithms, we need to store the successive constrained grids $O^{H_l}_{\eta_{l-1}}$ in an effective way. In particular, we need to determine if a candidate point $x \in X^{H_l}$ is in the constrained grid $O^{H_l}_{\eta_{l-1}}$, and access to any of its neighbors, in time $O(1)$, while keeping the storage to be in the order of the size of the (constrained) grids. Moreover, we need to store only the points of X^{H_l} that are in the set $O^{H_l}_{\eta_{l-1}}$. These sets are much smaller compared to the sets X^{H_l}, so we maintain them as a collection of nodes (hash table) accessed through a hash function, with a linked list to handle collisions. Each node of the constrained grid $O^{H_l}_{\eta_{l-1}}$ is instrumented with informations providing the value at this node, its position in the grid, and additional variables needed to maintain the set of active nodes in the fast marching propagation. For points of X^{H_l} that are not in the constrained grid $O^{H_l}_{\eta_{l-1}}$, the hash function returns an empty pointer.

5. COMPUTATIONAL COMPLEXITY

In order to choose the parameters of our algorithm, we shall assume that the following error estimation for the approximate value function $V^h_{\text{s}\to}$ holds for some $\theta > 0$:

$$|V^h_{\text{s}\to}(x) - v_{\text{s}\to}(x)| \leqslant Ch^\theta \ . \tag{6}$$

Indeed, by adapting the results of Capuzzo Dolcetta and Ishii (1984), using Assumption 2.1, one can obtain (6) with $\theta = 1/2$. If we further assume that f is of class $\mathcal{C}^2$, then we obtain (6) with $\theta = 1$.

So, our multi-level algorithm need to compute a numerical approximation with an error bound of same order in h, i.e., $\varepsilon \sim Ch^\theta$, for each level. In the two-level case, the two following parameters should be chosen:

1. The mesh step of the coarse grid H.
2. The parameter η^H, which should be big enough for O^H_η to contain the true optimal trajectories.

The estimation (6) give us a theoretical upperbound for η^H. Then, we have the following result:

Proposition 5.1. Assume there exists a finite number of optimal trajectories, and that the distance between $\mathcal{O}_\eta$ and Γ^* is in the order of η^β, with $0 < \beta \leqslant 1$. Then the total computational complexity of the two level fast marching algorithm is

$$\mathcal{C}(H, h) \sim \kappa((\frac{D}{H})^d + (\frac{(\eta^H)^\beta}{h})^{d-1}\frac{D}{h})$$

where D denotes the Euclidean distance from x_{src} to x_{dst}.

Note that the "stiffness" parameter β can take any value in $(0, 1]$, as shown in further work. Regarding the error estimation we want to have in fine grid, the parameter H has an optimal value, which we can compute easily.

Similarly, optimizing the parameters of the multi-level fast marching method, and using the error estimation for $V^h_{\text{s}\to}$, we get the following result:

Theorem 5.1. With same assumption as Proposition 5.1.

(i) In order to have an error bound ε, we shall take $h = C\varepsilon^2, H_l = C(H_{l+1})^{\frac{2^{l+1}-2}{2^{l+1}-1}}, \forall l \in \{2, 3, ..., N-1\}$. In this case the total computational complexity of our N-level algorithm is in the order of $(\frac{1}{\varepsilon})^{\frac{2}{1+\beta}d}$.

(ii) Suppose now $f \in \mathcal{C}^2$ or $\theta = 1$ in (6), and $\beta < 1$. Then, we shall take $h = C\varepsilon$ and $H_1 = Ch^{\frac{1}{N}}$, $H_l = C(H_{l+1})^{\frac{l}{l+1}}, \forall l \in \{1, 2, ..., N-1\}$. In this case, the total computational complexity of our N-level algorithm in the order of $(\frac{1}{\varepsilon})^{(1-\beta)d}$.

(iii) When $\beta = 1$, set $N = -\lceil d \log(h) \rceil$ and take $\{H_1, H_2, ..., H_{N-1}\}$ as proposed in (ii). Then, the total computational complexity of our algorithm reduces to $\kappa(-C^d d \log(\frac{1}{\varepsilon}))$.

We implememented the algorithm in C++, and ran it on a single core of a Quad-Core IntelCore I7 at 2.3Ghz, with 16Gb of memory. For an eikonal problem on a cubic domain, with obstacles, in dimension 4, with 5 grid levels, the algorithm ran in 59.2s, giving an error of 3.1%, to be compared with a time of 4241.9s for the classical fast marching method, with the same error. Classical fast marching ran out of time from dimension 5. Our algorithm solved a dimension 7 instance in 1300s, with 6 grid levels, leading to an accuracy of 7%.

6. CONCLUSION

We developed a multilevel fast marching algorithm, allowing one to solve an eikonal PDE by reducing the search space to neighborhoods of optimal trajectories. This yields improved complexity bounds, and solves examples of PDE up to dimension 7, for which it leads to a major speedup by comparison with ordinary grid based methods. We believe our method can be extended to HJB PDE of non-eikonal type. To do so, one needs to replace the fast marching sweeps by Bellman-type updates.

REFERENCES

Capuzzo Dolcetta, I. and Ishii, H. (1984). Approximate solutions of the bellman equation of deterministic control theory. *Applied Mathematics and Optimization*, 11(1), 161–181.

Fleming, W.H. and Soner, H.M. (2006). *Controlled Markov processes and viscosity solutions*, volume 25. Springer Science & Business Media.

Sanders, P. and Schultes, D. (2012). Engineering highway hierarchies. *Journal of Experimental Algorithmics (JEA)*, 17, 1–1.

Sethian, J.A. (1996). A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4), 1591–1595.