Équations de Hamilton-Jacobi

Méthode de Fast-Marching généralisée

T. Schmoderer N. Rouxelin A. Quinquennel D. Stambouli

Département Génie Mathématiques
INSA ROUEN

Présentation, 27 novembre 2017

Sommaire

Méthode classique

Équation eikonal

- Soit un domaine $\Omega \subset \mathbb{R}^2$ et
- Γ_t la modélisation du mouvement du front (courbe fermée)
- Le mouvement est dans la direction normale $\overrightarrow{n_{x,t}}$: $\frac{\partial \Gamma_t}{\partial t} = c(x) \cdot \overrightarrow{n_{x,t}}$
- Avec la méthode *level-set* Γ_t représente la ligne de niveau 0 d'une fonction u dans $\mathbb{R}^3$

Méthode classique

Équation eikonal

- Ω_t le domaine de R^2 délimité par Γ_t
- $\forall t \in \mathbb{R}_+,\ \forall x \in \mathbb{R}^2, x \in \Omega_t \Rightarrow u(x,t) > 0$
  $x \in \Omega_t^c \Rightarrow u(x,t) < 0$ et $u(\Gamma_t, t) \equiv 0$
- $$\frac{\mathrm{d}u}{\mathrm{d}t}(x(t),t) = 0, \forall x \in \Gamma_t$$
- Équation eikonal $\frac{\partial u}{\partial t}(x,t) = c(x)\,|\nabla_x u(x,t)|$

Méthode classique

La méthode fast-marching classique

- Cette méthode est valable seulement si la vitesse est positive
- Soit $T(x)$ le temps d'arrivée du front au point x, on veut $u(x,t) = T(x) + t$
- L'équation eikonal devient donc : $1 = c(x)|\nabla T|$

Méthode classique

Schéma numérique en T

- $x_{ij} := (i\Delta x, j\Delta x) \qquad (i,j) \in \mathbb{Z}^2$ avec $\Delta x > 0$
- Soit $T_{i,j}$ l'approximation de T au point x_{ij}
- Approximation du gradient par un schéma différences finies
- $\max\left(\frac{T_{ij}-T_{i+1,j}}{\Delta x}, \frac{T_{ij}-T_{i-1,j}}{\Delta x}, 0\right)^2 + \max\left(\frac{T_{ij}-T_{i,j-1}}{\Delta x}, \frac{T_{i,j}-T_{i,j+1}}{\Delta x}, 0\right)^2 = \frac{1}{c_{ij}^2}$

Méthode classique

Algorithme

- *frozen points* : À l'instant t_n, on note Fr^n l'ensemble des points visités par le front dont la valeur de T est connue
- La *narrow band* : il s'agit du voisinnage des *frozen points*, c'est-à-dire les points non visités mais qui ont un voisin déhà visité. À l'instant t_n, on la note NB^n.
- Les points éloignés : ce sont tous les autres points du domaine

Méthode classique

Algorithme

	Initialisation	
Matrice de passage :	$T_{ij} = 0$	$\forall (i,j)$ tq $(i\Delta x, j\Delta x) \in \Omega_0$
Temps :		$n = 0$
	Boucle	
1. Calculer T_{ij}		$\forall (i,j) \in NB^n$
2. Mettre à jour Fr^{n+1} :	Accepter les (i,j)	tq $T_{i,j} = \min_{(a,b)} T_{a,b}$
3. Mettre à jour NB^{n+1}	:	Frontière de Fr^{n+1}
4. Mettre à jour n	:	$n = n + 1$

Application

Exemple 1

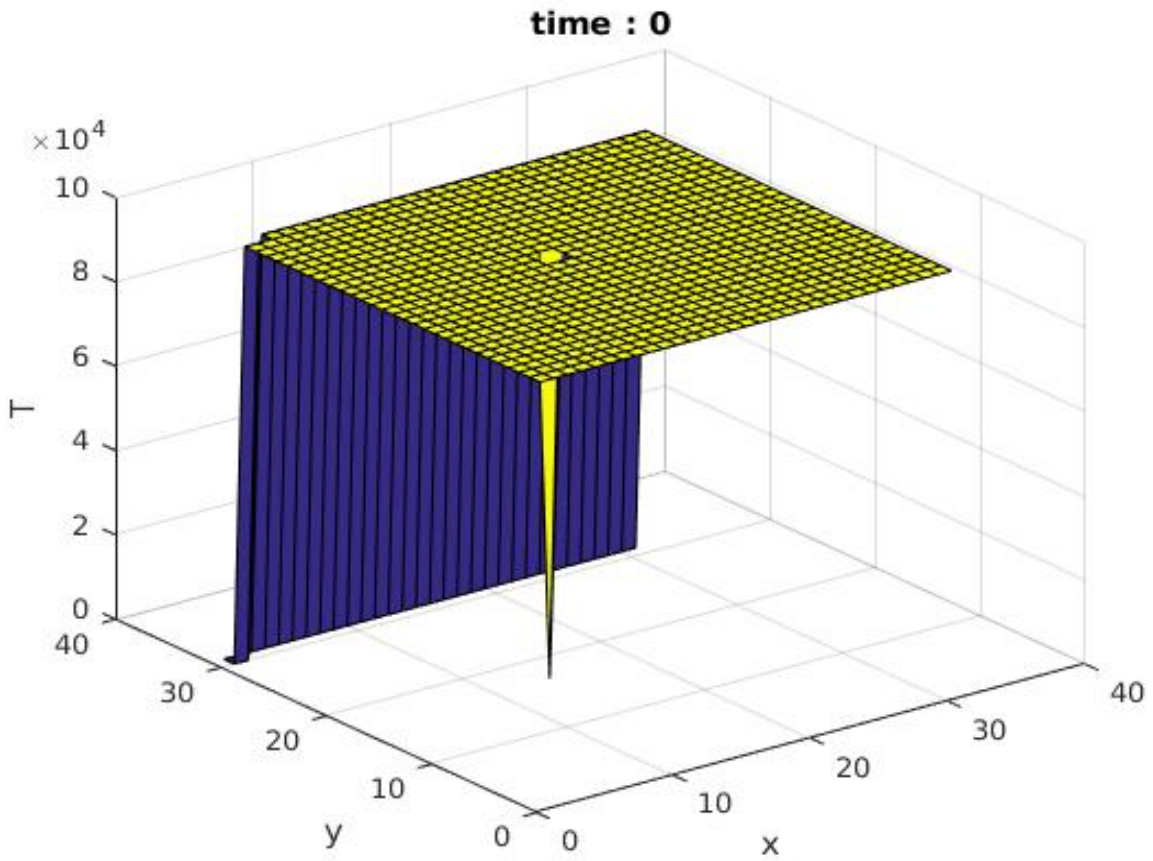


Figure – Exemple 1 : front initial

Application

Exemple 1

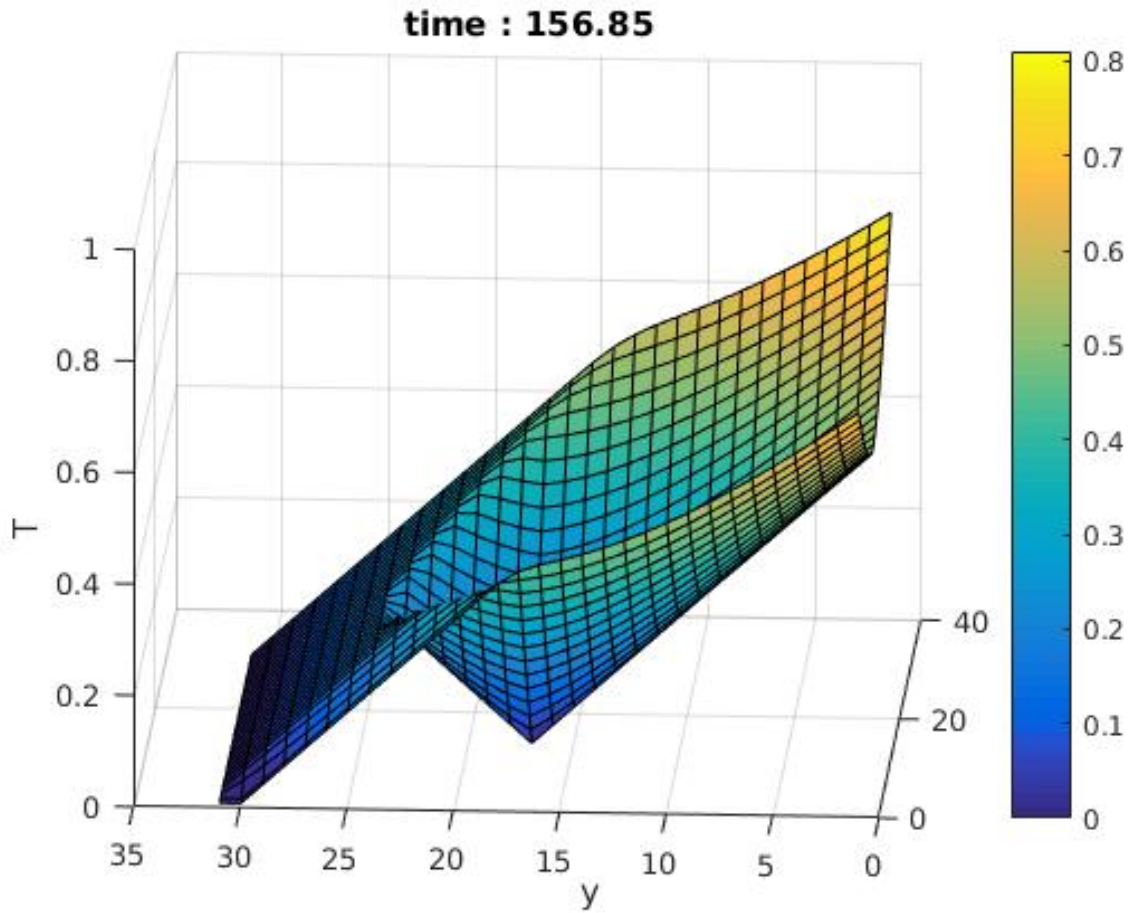


Figure – Exemple 1 : front final

Application

Exemple 1

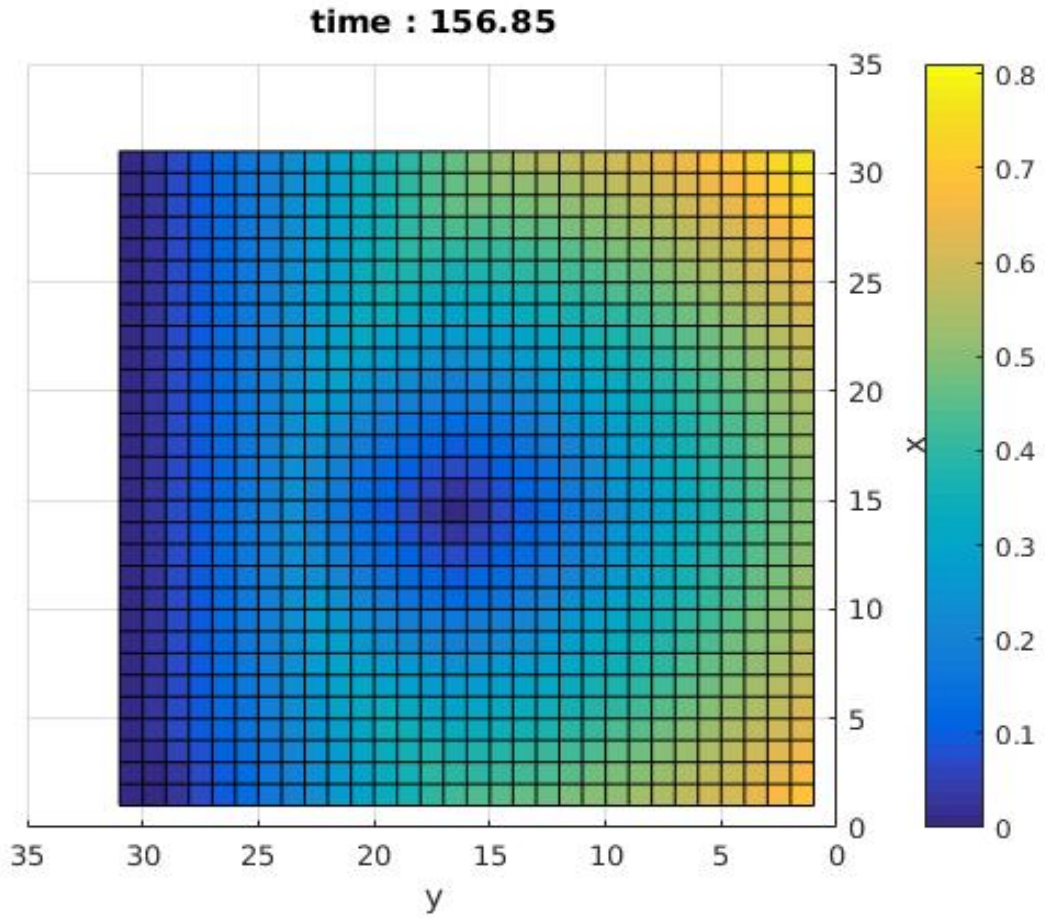


Figure – Exemple 1 : front final vu du dessus

Application

Exemple 2

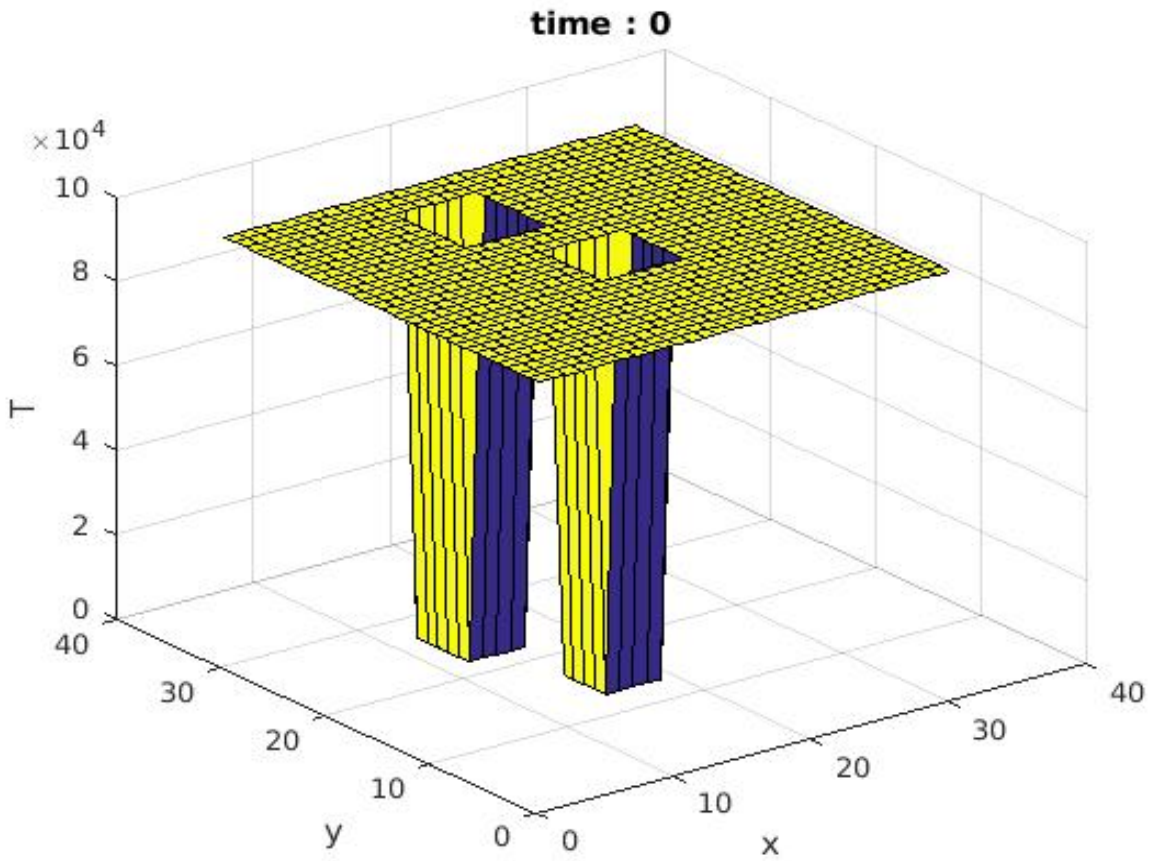


Figure – Exemple 2 : front initial

Application

Exemple 2

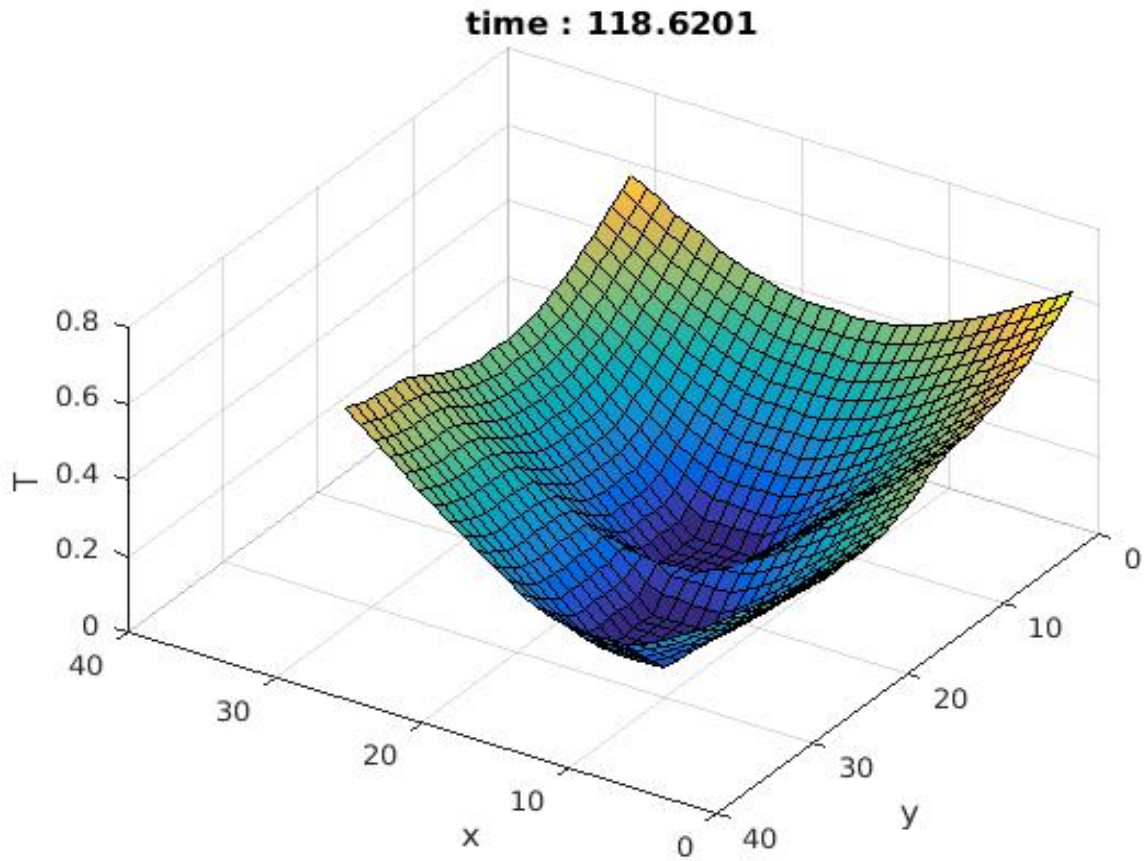


Figure – Exemple 2 : front final

Application

Exemple 2

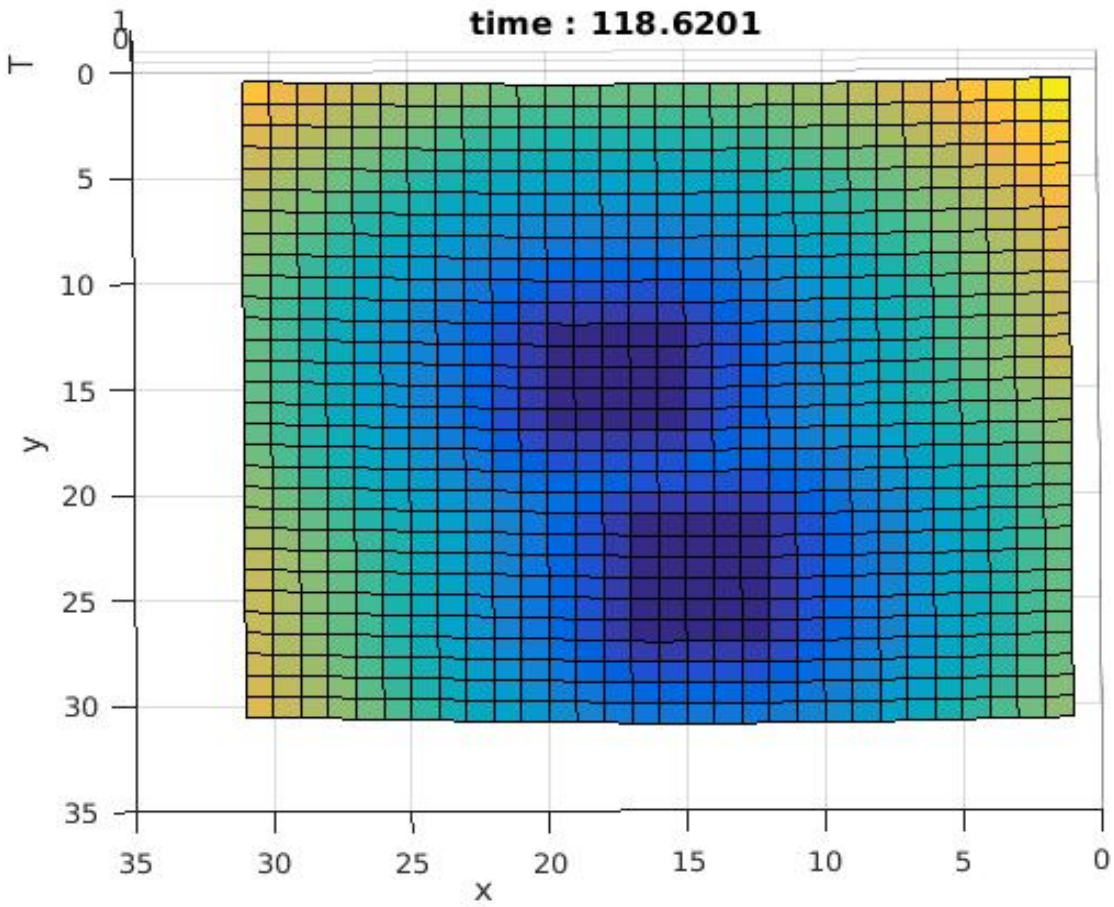


Figure – Exemple 2 : front final vu du dessus

La méthode *fast-marching* généralisée

Re-définition du problème

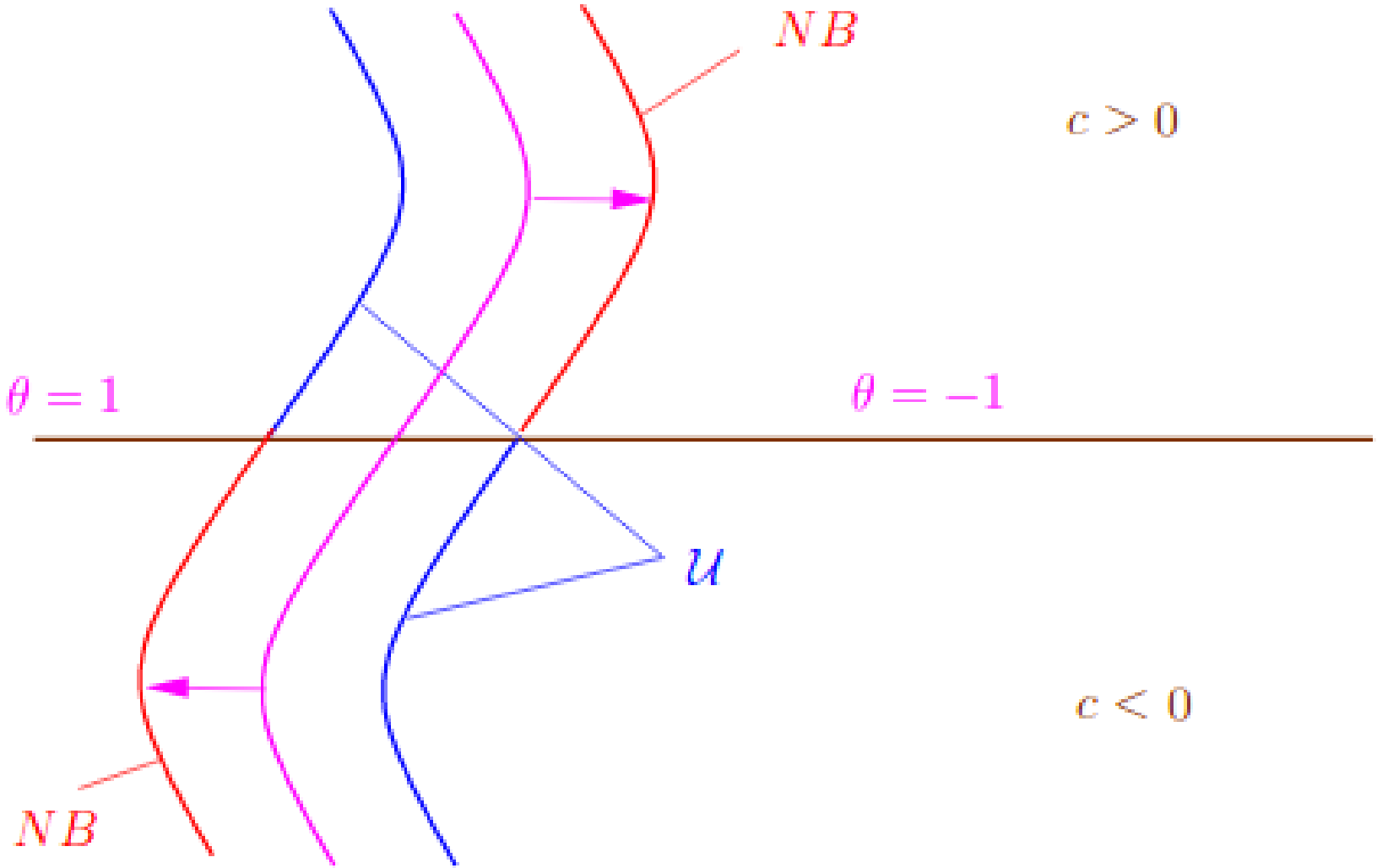
- On vise à généraliser pour traiter les vitesses quelqconques : on introduit le champ θ

$$\begin{cases} \theta = 1 & \text{dans } \Omega_t \\ \theta = -1 & \text{dans } \Omega_t^c \end{cases}$$

- On ne parle plus de *frozen points* puisque le front peut passer plusieurs fois sur un point : un introduit une *frozen region* à l'itération n, notée Fr^n
- On redéfinit la *narrow band* comme le voisinage de la *frozen region*, directement atteignable par le front, notée NB^n
- On définit les points utiles à un point $I \in NB^n$ comme les points $\in V(I)$ et $\in Fr^n$, notés $\mathcal{U}^n(I)$
- On note U^n l'ensemble des points utiles à NB^n

La méthode *fast-marching* généralisée

Re-définition du problème



La méthode *fast-marching* généralisée

Re-définition du problème

- On introduit une régularisation de la vitesse :

$$\hat{c}_I^n \equiv \begin{cases} 0 & \text{si il existe } J \in V(I) \text{ tel que } c_I^n c_J^n < 0 \text{ et } |c_I^n| < |c_J^n| \\ c_I^n & \text{sinon} \end{cases}$$

- Pour s'assurer d'utiliser uniquement les points utiles à I pour le calcul de la valeur temporaire $\tilde{T}_I^n$, on introduit

$$T_{J\to I}^n = \begin{cases} T_J^n & \text{si } J \in \mathcal{U}^n(I) \\ +\infty & \text{sinon} \end{cases}$$

- L'algorithme repose sur la résolution de l'équation suivante ($I = (i_1, i_2)$

$$\max\left(0, \tilde{T}_I^{n-1} - T_{(i_1+1,i_2)\to I}^{n-1}, \tilde{T}_I^{n-1} - T_{(i_1-1,i_2)\to I}^{n-1}\right)^2 +$$

$$\max\left(0, \tilde{T}_I^{n-1} - T_{(i_1,i_2+1)\to I}^{n-1}, \tilde{T}_I^{n-1} - T_{(i_1,i_2-1)\to I}^{n-1}\right)^2 = \frac{(\Delta x)^2}{|\hat{c}_I^{n-1}|^2}$$

Méthode *fast-marching* généralisée

Initialisation

Temps : $n = 1, t_0 = 0$

θ_0 : $\theta^0 = \begin{cases} 1 & \text{si } x_I \in \Omega_0 \\ -1 & \text{sinon} \end{cases}$

T_0 : $T_I^0 = \begin{cases} 0 & \text{si } I \in \mathcal{U}^0 \\ +\infty & \text{sinon} \end{cases}$

Boucle

1. Calculer $\tilde{T}_I^{n-1}$: Résoudre l'équation $\forall I \in NB^{n-1}$
2. Calcul du temps : $\tilde{t}_n = \inf_{I \in NB^{n-1}} \tilde{T}_I^{n-1}$

Tronquer $\tilde{t}_n$: $t_n = \max\left(t_{n-1}, \min\left(\tilde{t}_n, t_{n-1} + \Delta t\right)\right)$

Si $t_n = t_{n-1} + \Delta t$ et $t_n < \tilde{t}_n$: Revenir à 1 avec $n = n+1$, $\theta^n = \theta^{n-1}$ et $T^n = T^{n-1}$

3. Mettre à jour NA^n : $NA^n = \left\{ I, \tilde{T}_I^{n-1} = \tilde{t}_n \right\}$
4. Mettre à jour θ^n : $\theta_I^n = \begin{cases} -\theta_I^{n-1} & \text{si } I \in NA^n \\ \theta_I^{n-1} & \text{sinon} \end{cases}$
5. Calcul de T_I^n : $T_I^n = \begin{cases} t_n & \text{si } I \in NA^n \text{ et } I \in \mathcal{U}^n \\ T_I^{n-1} & \text{si } I \in \mathcal{U}^{n-1} \backslash NA^n \text{ et } I \in \mathcal{U}^n \\ +\infty & \text{sinon} \end{cases}$
6. Mettre à jour n : $n = n+1$

Démonstration

Bibliographie I


N.Forcadel, C.Le Guyader, C.Gout
Generalized fast marching method : applications to image segmentation.