Algorithms for globally optimal autonomous vehicle paths

by Blane Rhoads
for Intel's Computational Lithography Group
Friday, May 10th, 2013

Overview

About me:

- Mechanical Engineering, Applied Math, "Dynamical Systems, Controls, and Robotics", "Computational Science and Engineering".
- ODEs/PDEs, data visualization, spatial data structures, optimization, and algorithm/software development.
- Intel, Sandia National Lab, United Technologies Research Center.
- Excited to transition from academia to industry, to work in a team environment, and to learn.

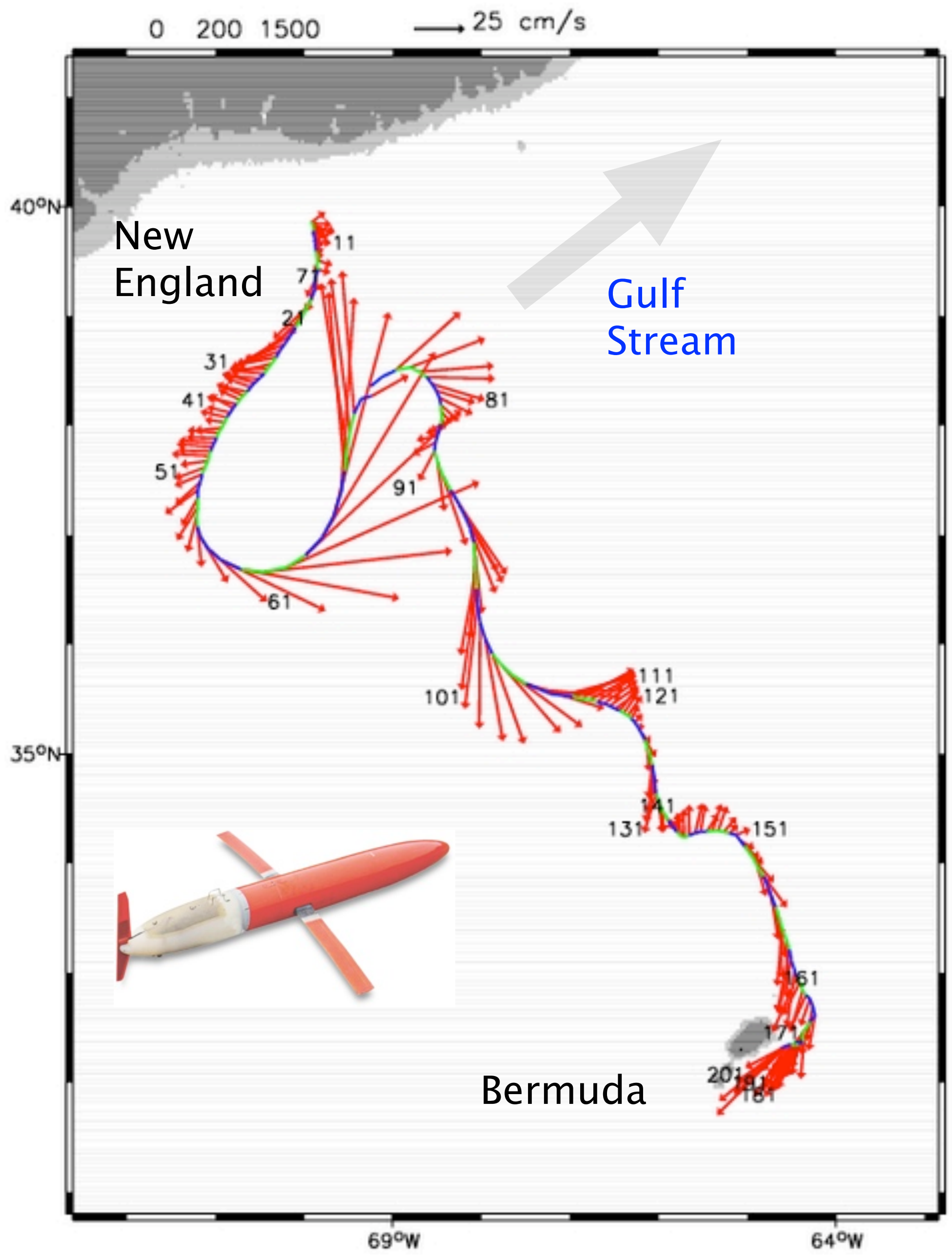
About this talk:

- Numerical methods for optimal trajectories of low-dimensional but *spatially complex, time-varying, nonlinear systems*.
- Focus is on *front-tracking methods for global optima*, rather than iterative methods for local optima, with emphasis on (1) min time via particle methods, and (2) min energy via backwards time-stepping.
- Themes/keywords: shock-capturing, controllability/reachability, remeshing and trimming in the particle method, adaptive (quadtree/octree/triangle) grids.

Motivation: underwater gliders

Pro: range, duration, stealth
Con: 25-40 cm/s
(0.9-1.4 kph)

Actual 50-day trajectory (blue) and depth-averaged currents (red) [spray.ucsd.edu]



Minimum time problem

$$\min_{t_f,\mathbf{u}} \int_{t_0}^{t_f} (1)dt$$ subject to:

$$\dot{\mathbf{x}} = \mathbf{v}(\mathbf{x}, t) + \mathbf{u}$$

2D, time-varying vector flow field

$$\mathbf{x}(t_0) = \mathbf{x}_0$$

$$\mathbf{x}(t_f) = \mathbf{x}_f$$

$$t_f \in [t_0, t_0 + T_{\max}]$$

$$|\mathbf{u}| \leq s$$

Minimum time problem

$$T(\mathbf{x}_0, t_0) := \min_{t_f,\mathbf{u}} \int_{t_0}^{t_f} (1)dt$$ subject to:

$$\dot{\mathbf{x}} = \mathbf{v}(\mathbf{x}, t) + \mathbf{u}$$

2D, time-varying vector flow field

$$\mathbf{x}(t_0) = \mathbf{x}_0$$

$$\mathbf{x}(t_f) = \mathbf{x}_f$$

$$t_f \in [t_0, t_0 + T_{\text{max}}]$$

$$|\mathbf{u}| \leq s$$

Minimum time problem

$$T(\mathbf{x}_0, t_0) := \min_{t_f, \mathbf{u}} \int_{t_0}^{t_f} (1)\, dt$$ subject to:

$$\dot{\mathbf{x}} = \mathbf{v}(\mathbf{x}, t) + \mathbf{u}$$

$$\uparrow$$

2D, time-varying
vector flow field

$$\mathbf{x}(t_0) = \mathbf{x}_0$$

$$\mathbf{x}(t_f) = \mathbf{x}_f$$

$$t_f \in [t_0, t_0 + T_{\max}]$$

$$|\mathbf{u}| \leq s$$

T = solution of Hamilton Jacobi Bellman (HJB) PDE.

Feedback control law $$\mathbf{u} = -s \frac{\nabla T}{|\nabla T|}$$

Why is this problem difficult?

- Spatial complexity of **v** (*Shocks* in *T*)
- |**v**| > s (*Discontinuities* in *T*)
- Time-dependence of **v**

Benchmark for **v**(**x**,t) = 0: Fast Marching (FM) method

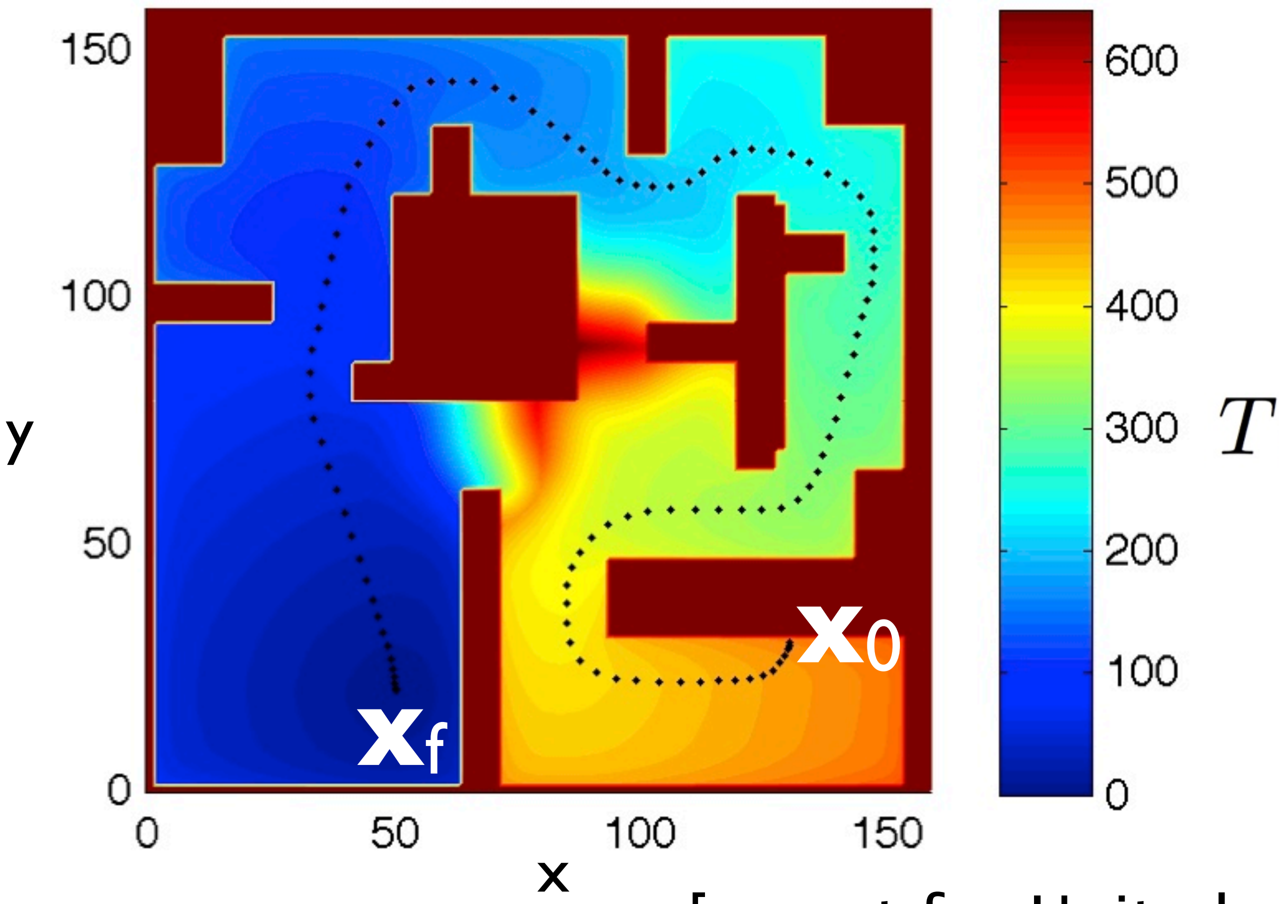
Front tracking method

HJB = Eikonal:

$$s\,|\nabla T| = 1$$

Boundary condition:

$$T(\mathbf{x}_f) = 0$$



[report for United Technologies (2011)]

Benchmark for **v**(**x**,t) = 0: Fast Marching (FM) method

Front tracking method

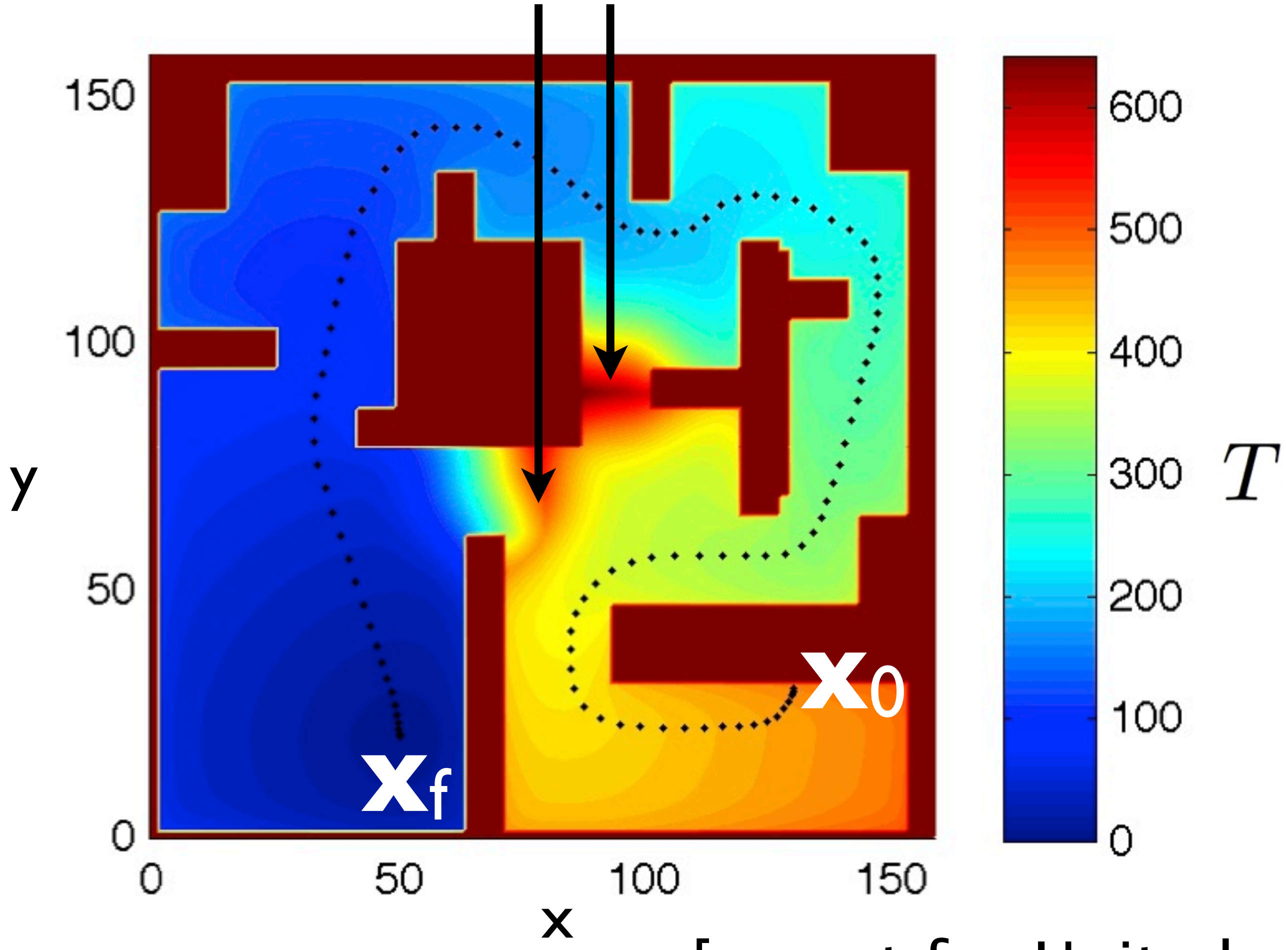
HJB = Eikonal:

$$s\,|\nabla T| = 1$$

Boundary condition:

$$T(\mathbf{x}_f) = 0$$

shocks (ridges) in T



[report for United Technologies (2011)]

Benchmark for **v**(**x**,t) = **v**(**x**): Ordered Upwind Methods (OUMs)

Extension of FM method

HJB:

Limitation: Can’t handle |**v**| > s.

Benchmark for **v**(**x**,t) = **v**(**x**): Ordered Upwind Methods (OUMs)

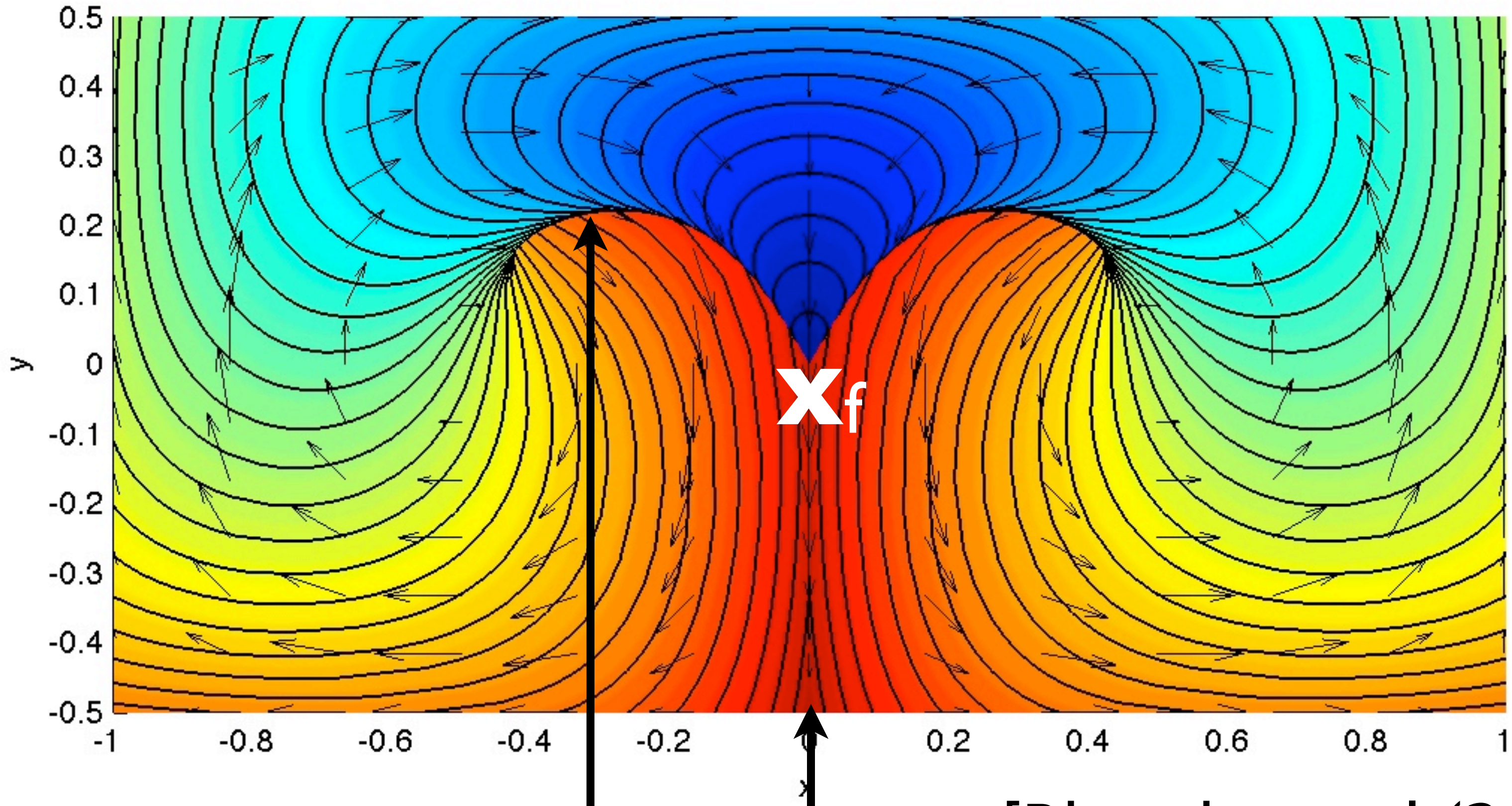
Extension of FM method

HJB:

$$\underbrace{\left(s - \mathbf{v}(\mathbf{x}) \cdot \frac{\nabla T}{|\nabla T|}\right)}_{\text{speed of front}} |\nabla T| = 1$$

Limitation: Can't handle |**v**| > s.

Alternative (for |**v**| > s): particle method



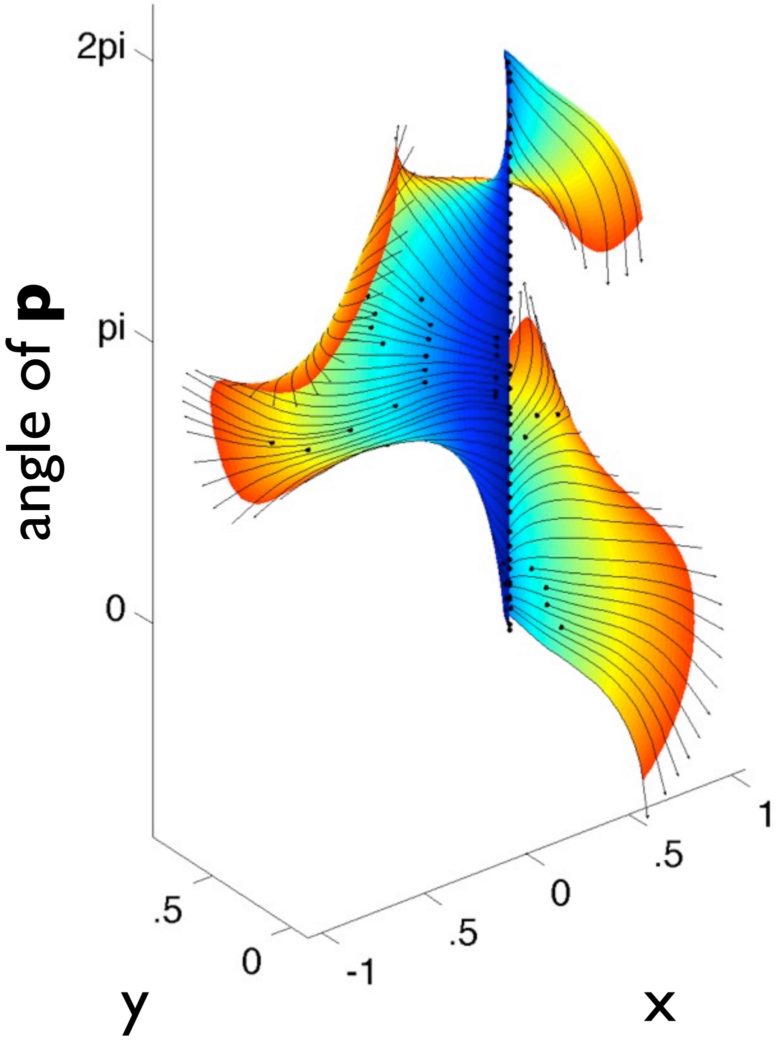
[Rhoads, et al (2010)]

discontinuity

shock

Key to particle method: track state **x** *and costate* ***p*** (vector normal to front)

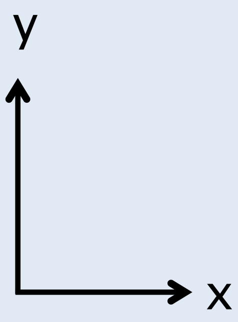
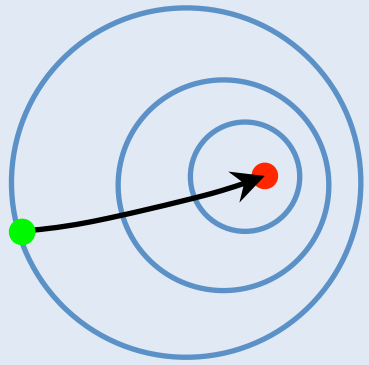
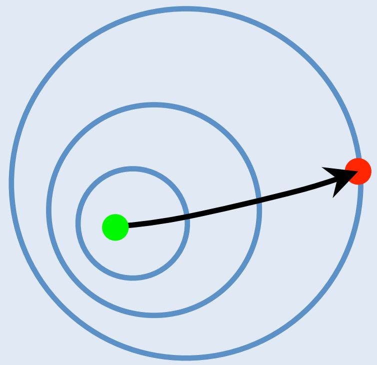
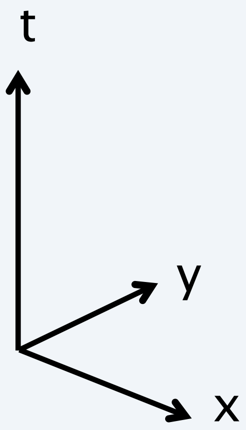
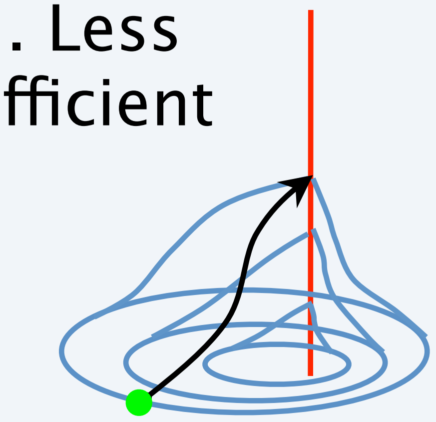
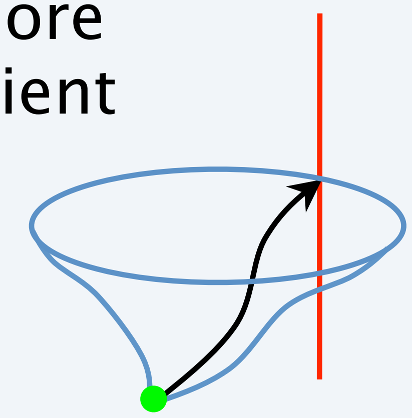
$$\dot{\mathbf{p}} = -\nabla \mathbf{v}^{\mathrm{T}} \mathbf{p}$$



Theorem:
It's sufficient to track the angle of **p**

"extremal surface"
= continuum of fronts
= continuum of trajectories

Extends to time-varying case (w/ 2 options)

legend: $\mathbf{x}_0$ 🟢 $\mathbf{x}_f$ 🔴	**backward: "controllability"** **feedback, but...**	**forward: "reachability"** **no feedback, but...**
time-invariant case; $\mathbf{v}(\mathbf{x},t) = \mathbf{v}(\mathbf{x})$ 		
time-varying case 	... Less efficient  [Rhoads, et al (2010)]	... More efficient  [Rhoads, et al (2013)]

Forward, time-varying particle method
Ex: Adriatic Sea (s = 0.9 kph, T_{max} = 54 h)

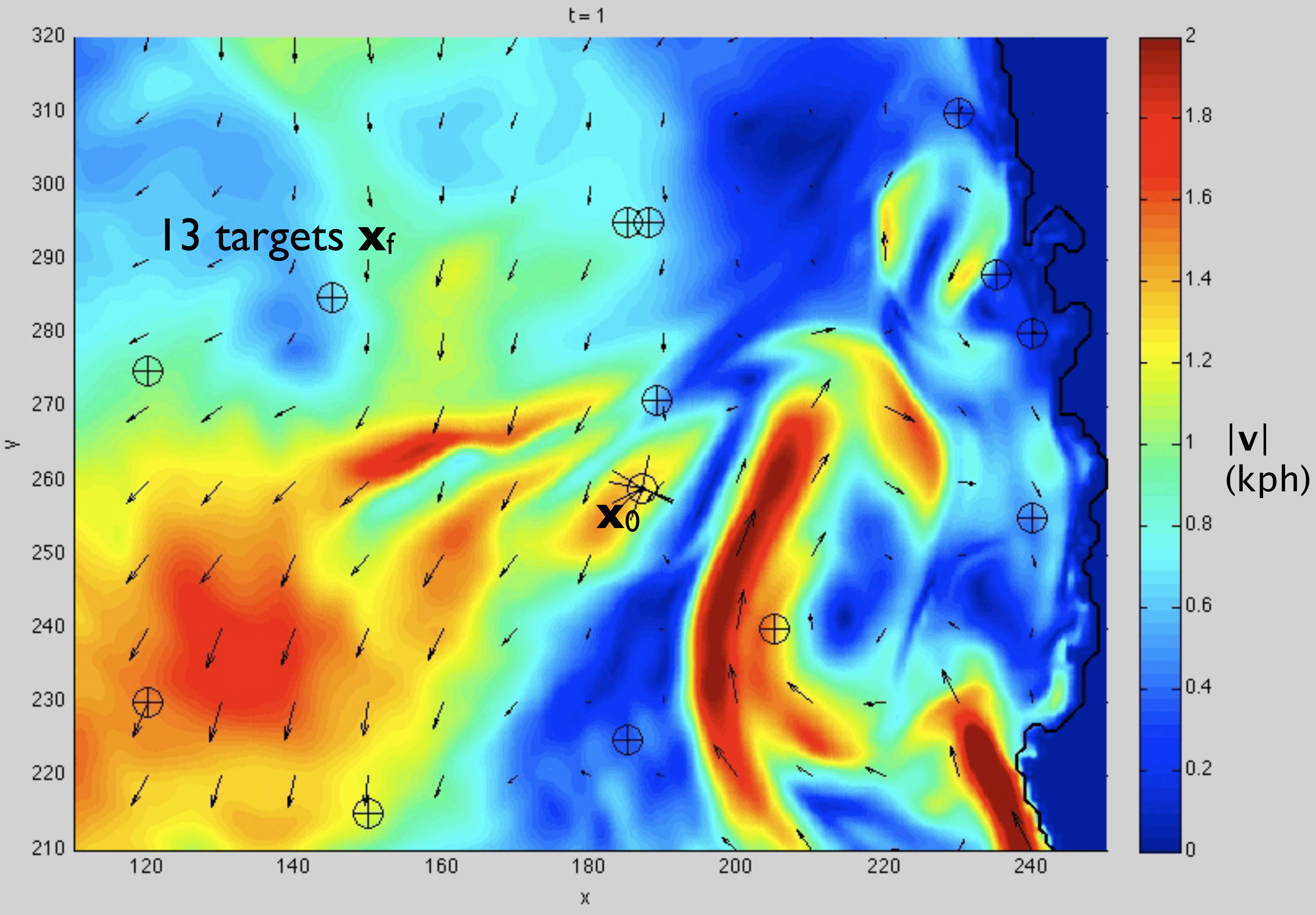
13 targets $\mathbf{x}_f$

$|\mathbf{v}|$
(kph)

$\mathbf{x}_0$

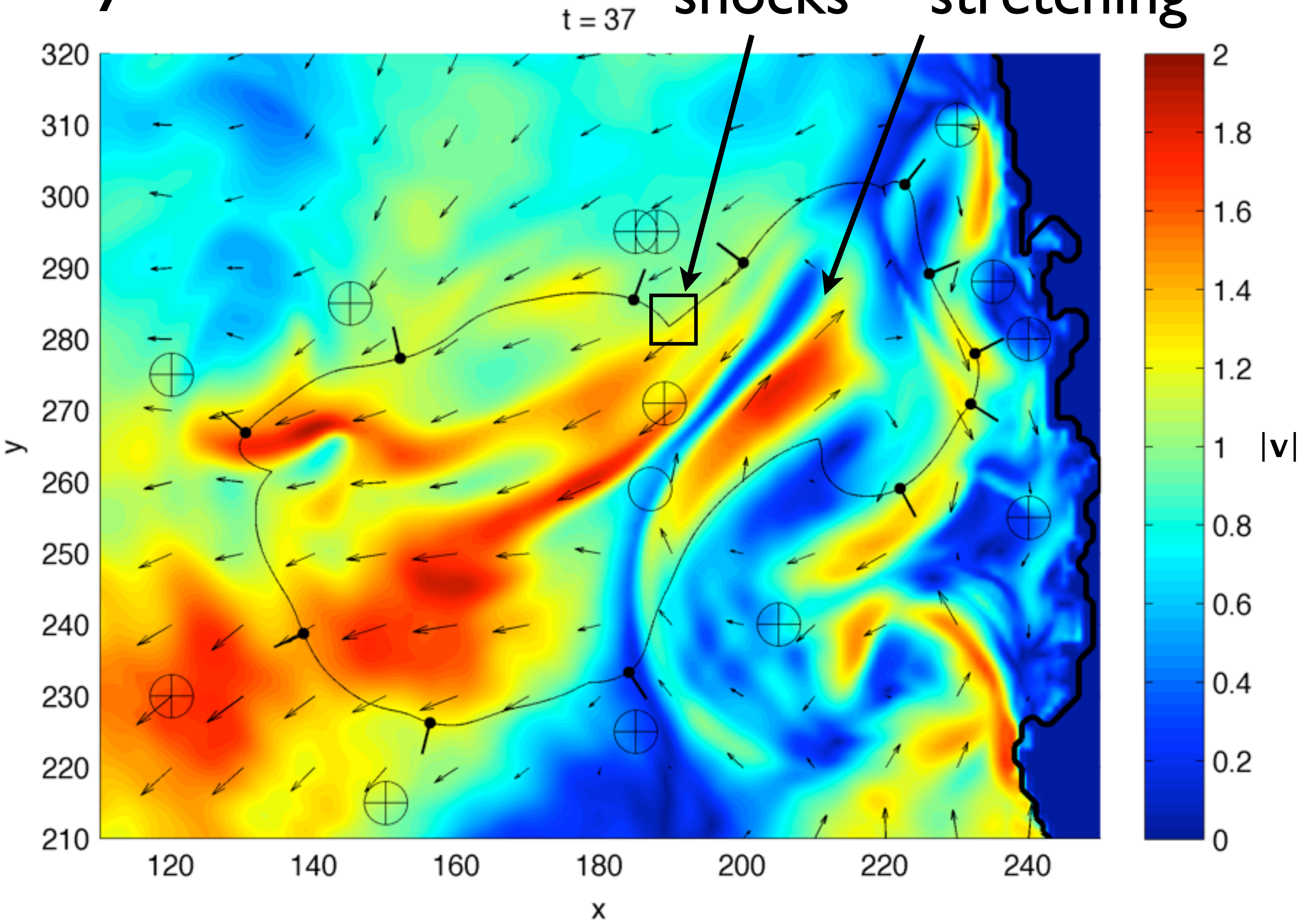
Movie of “reachability” front [Rhoads, et. al, 2013]

Forward, time-varying particle method
Ex: Adriatic Sea (s = 0.9 kph, T_{max} = 54 h)



Movie of “reachability” front [Rhoads, et. al, 2013]

Two key features

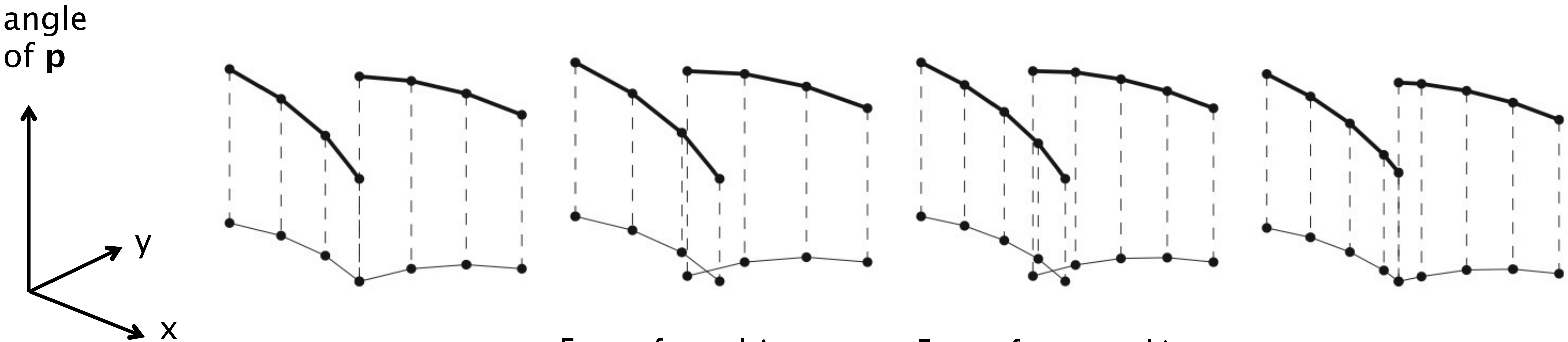


shocks

stretching

Snapshot of “reachability” front [Rhoads, et. al, 2013]

Two key ingredients: *trimming* and *remeshing*



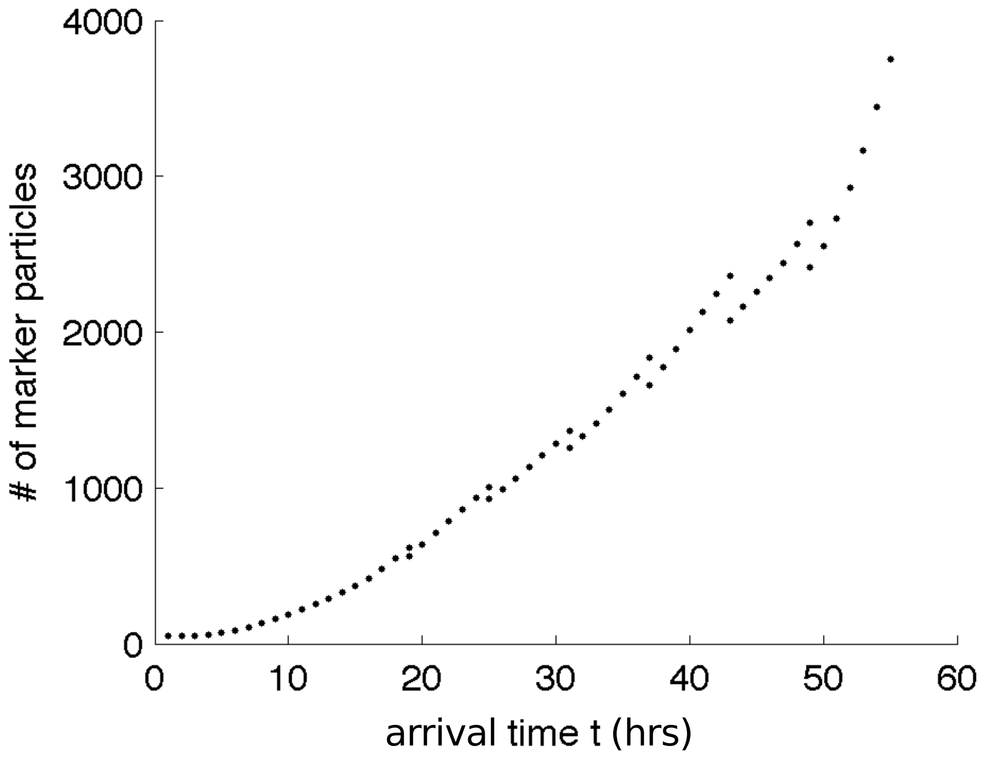
Front at time t

Front after solving ODEs

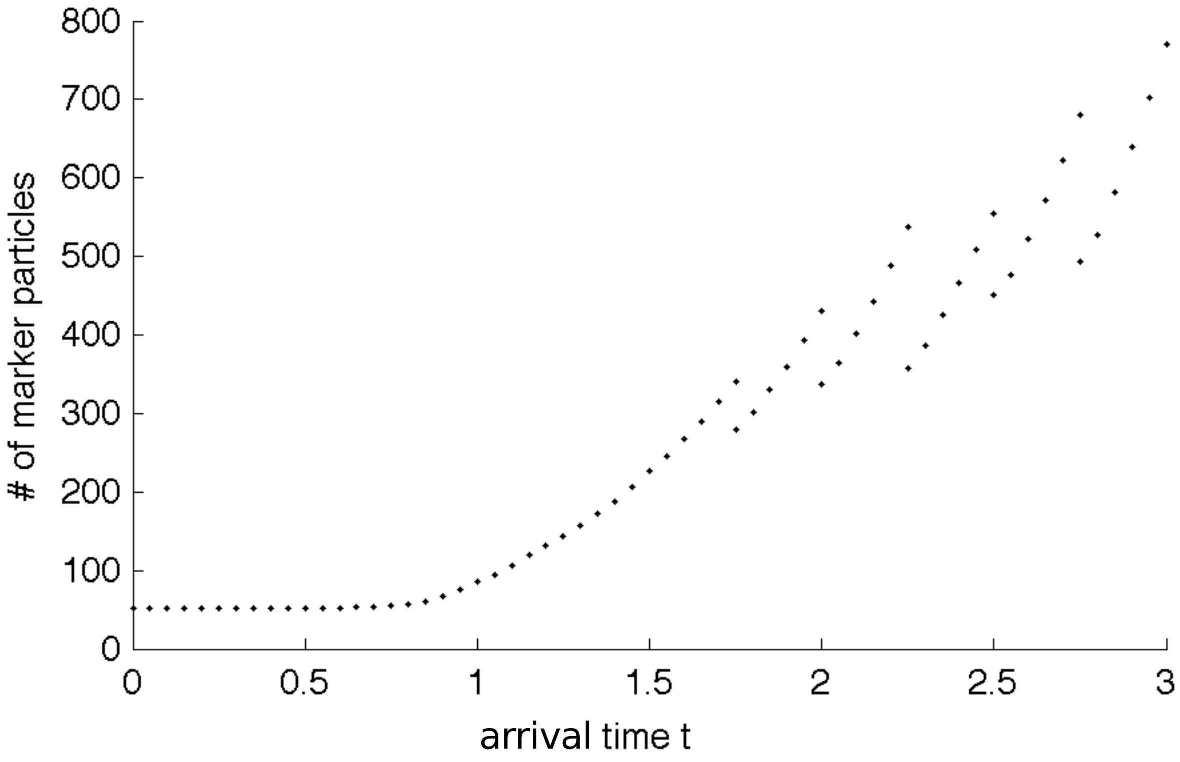
Front after remeshing (insertion/ redistribution of particles)

Front after trimming

Trimming adds complexity but also efficiency.

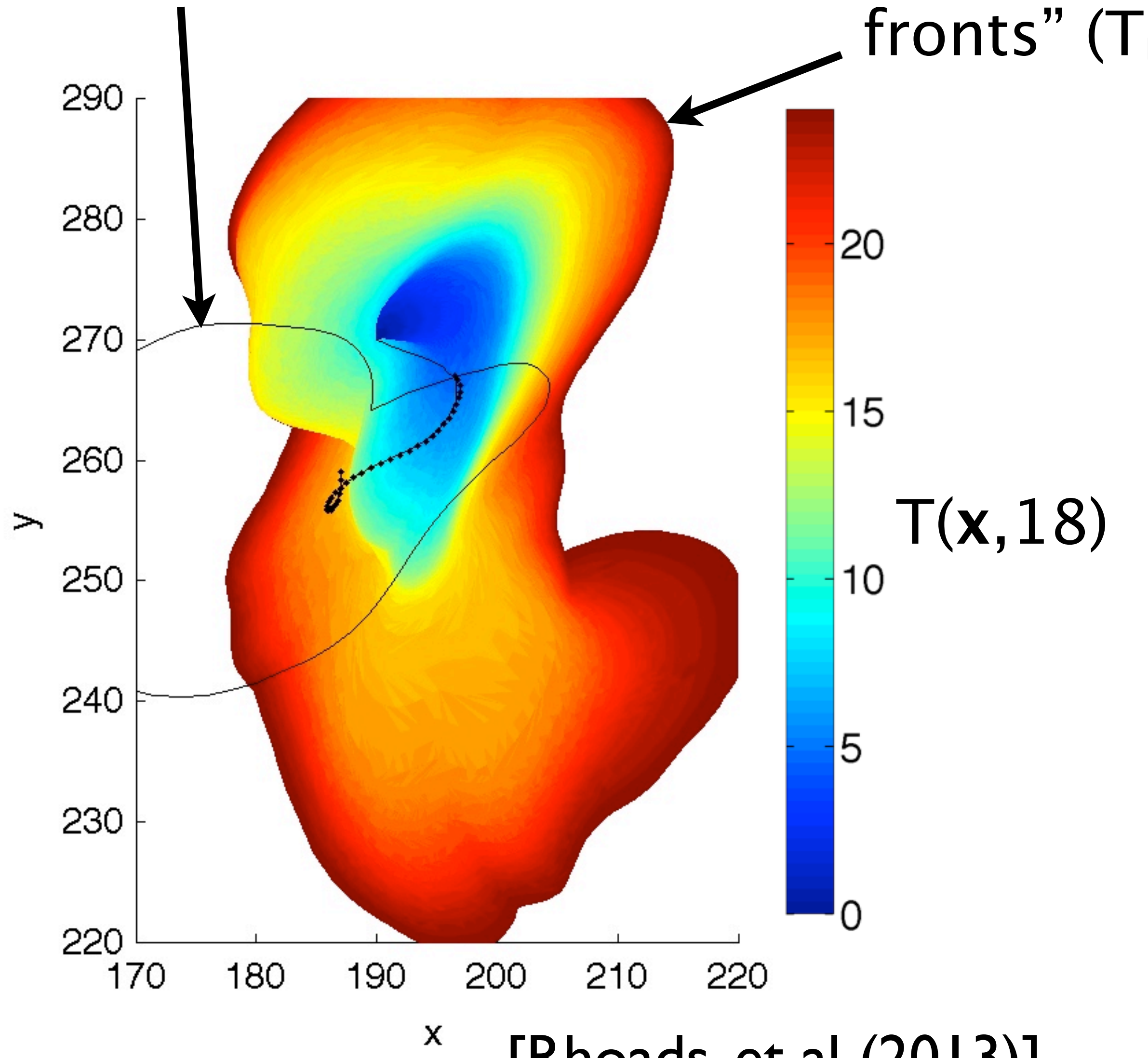


Present example (Adriatic)
(coast line decreases efficiency)



Another example (time-invariant gyre flow). Growth effectively linear instead of exponential.

Validation



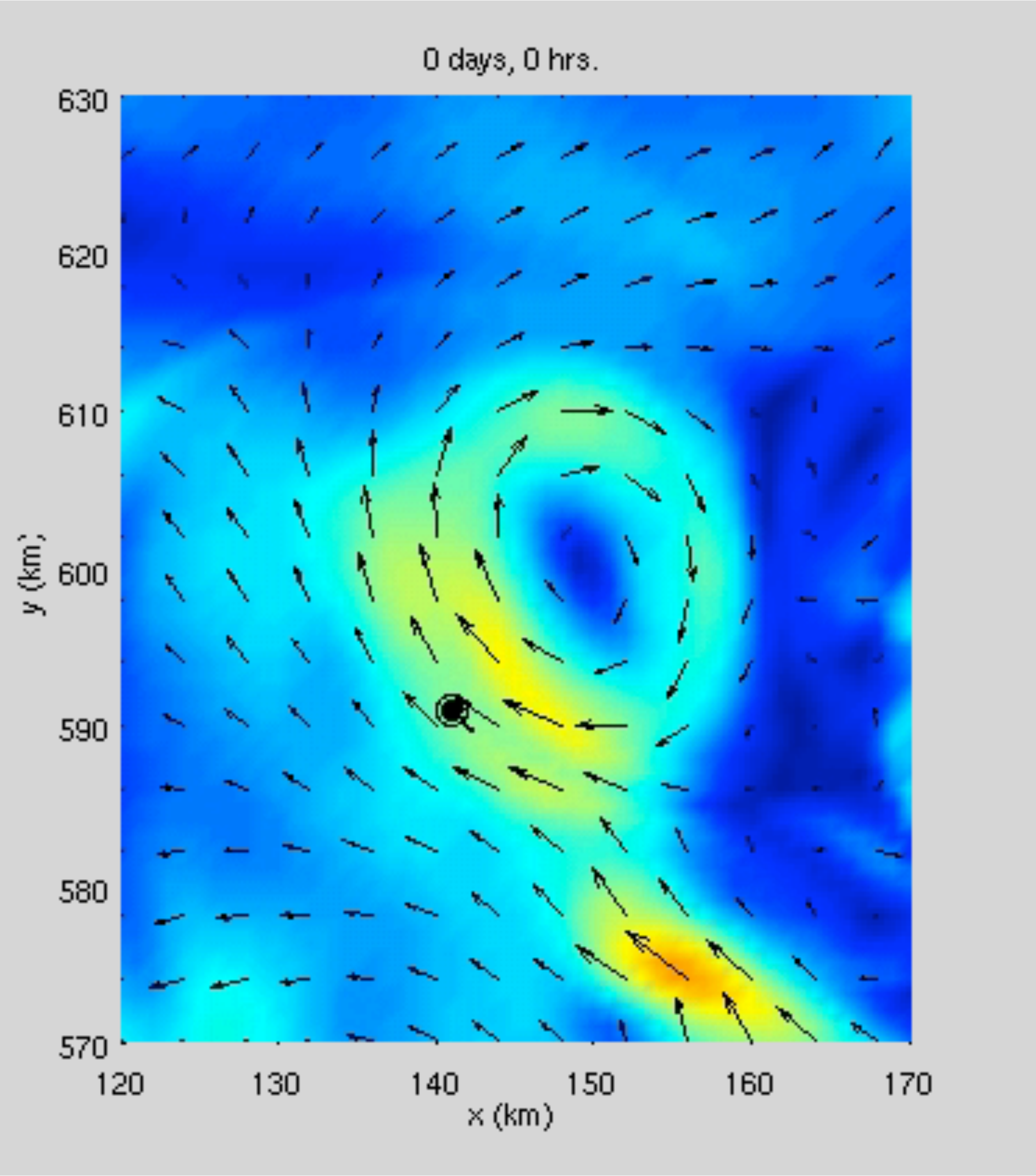
Forward result:
“reachability front”
at time t = 18 h

Backward result:
many
“controllability
fronts” (T_{max} = 24 h)

$T(\mathbf{x},18)$

[Rhoads, et al (2013)]

Example application:
approximate station keeping (Adriatic)



One vehicle treating flow as unknown disturbance

Other vehicle using flow forecast and min time control computed by forward method

Color indicates flow strength (in kph). Red ~ 2 kph (vehicle s = 0.9 kph).

Minimum energy / end cost problem

t_f fixed
no hard constraint on $\mathbf{x}(t_f)$

Cost-to-go $$V(\mathbf{x}_0, t_0) := \min_{\mathbf{u}} \left\{ \overset{\substack{\text{end cost}\\ \downarrow}}{h}(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} \overset{\substack{\text{weight on energy}\\ \downarrow}}{W} |\mathbf{u}|^2 dt \right\}$$

Minimum energy / end cost problem

t_f fixed
no hard constraint on $\mathbf{x}(t_f)$

Cost-to-go $$V(\mathbf{x}_0, t_0) := \min_{\mathbf{u}} \left\{ \overset{\substack{\text{end cost}\\ \downarrow}}{h}(\mathbf{x}(t_f)) + \int\limits_{t_0}^{t_f} \overset{\substack{\text{weight on energy}\\ \downarrow}}{W} |\mathbf{u}|^2 dt \right\}$$

$$\downarrow$$

HJB: $$\frac{\partial V}{\partial t} + \min_{\mathbf{u}} \left\{ (\mathbf{v} + \mathbf{u}) \cdot \nabla V + W|\mathbf{u}|^2 \right\} = 0$$

Boundary condition: $$V(\mathbf{x}, t_f) = h(\mathbf{x})$$

Minimum energy / end cost problem

t_f fixed

no hard constraint on $\mathbf{x}(t_f)$

Cost-to-go $$V(\mathbf{x}_0, t_0) := \min_{\mathbf{u}} \left\{ \overset{\substack{\text{end cost}\\ \downarrow}}{h}(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} \overset{\substack{\text{weight on energy}\\ \downarrow}}{W} |\mathbf{u}|^2 dt \right\}$$

↓

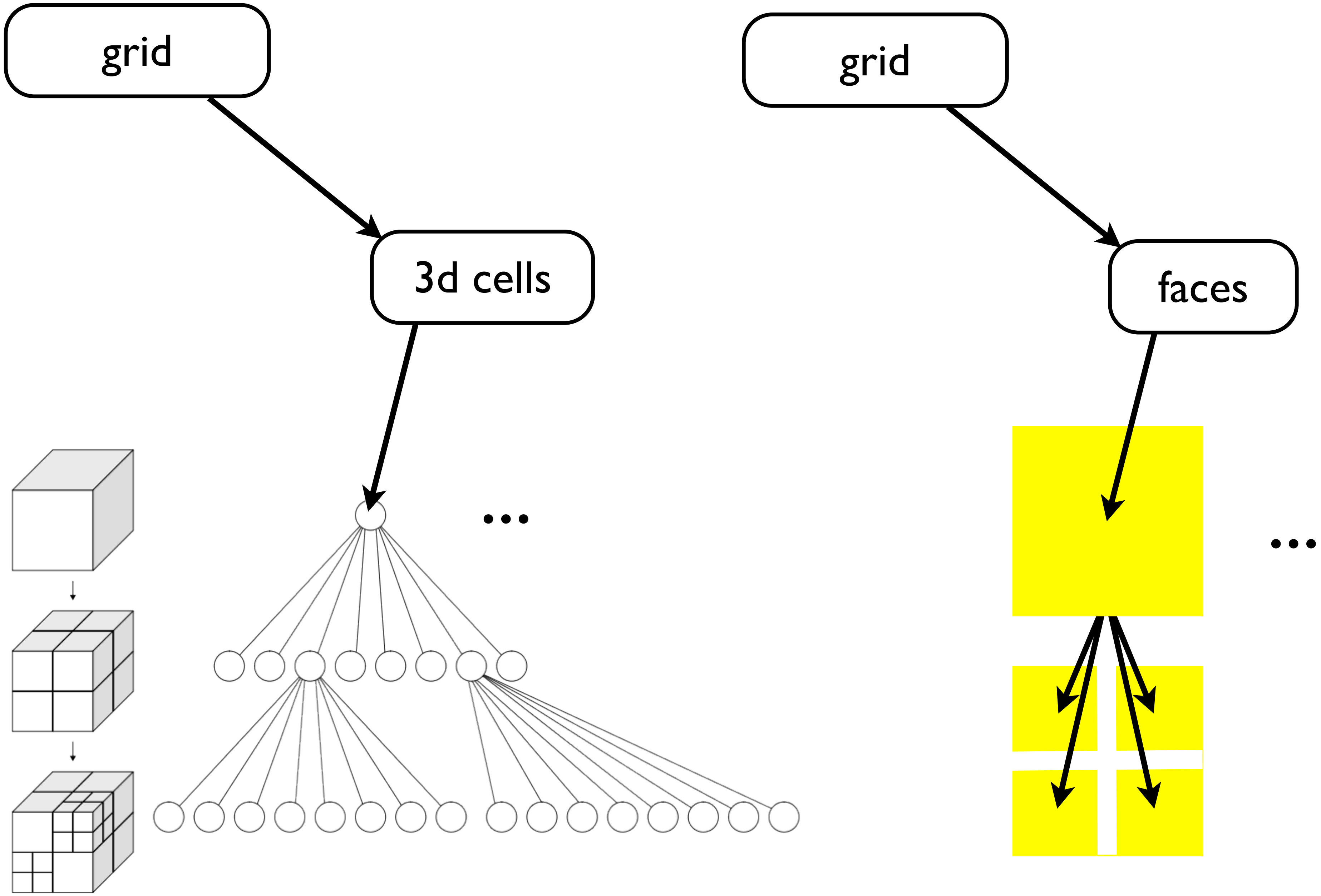
HJB: $$\frac{\partial V}{\partial t} + \min_{\mathbf{u}} \left\{ (\mathbf{v} + \mathbf{u}) \cdot \nabla V + W |\mathbf{u}|^2 \right\} = 0$$

Boundary condition: $$V(\mathbf{x}, t_f) = h(\mathbf{x})$$

↓

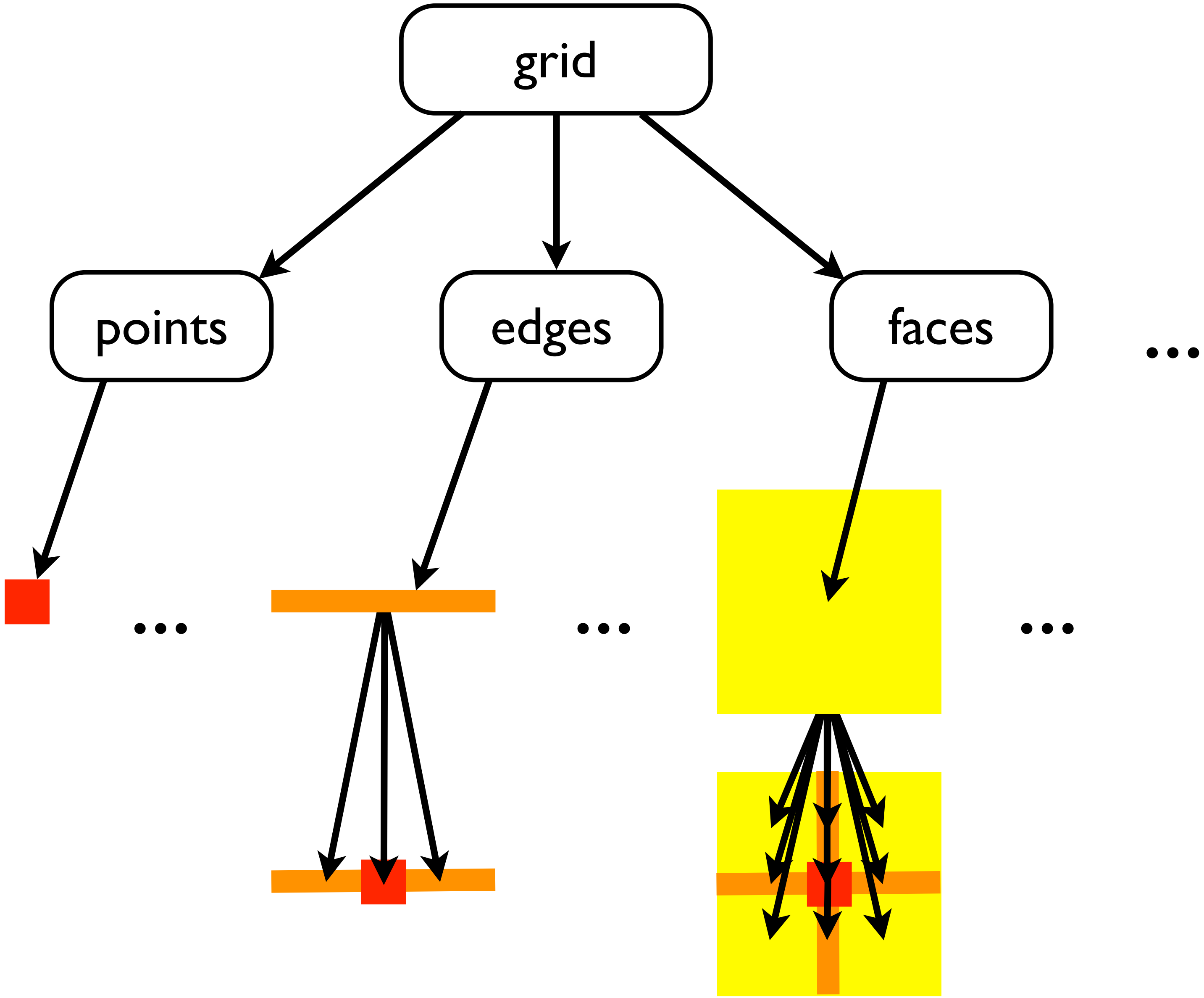
Backwards time-stepping

Adaptive grid structures: basic octree, quadtree



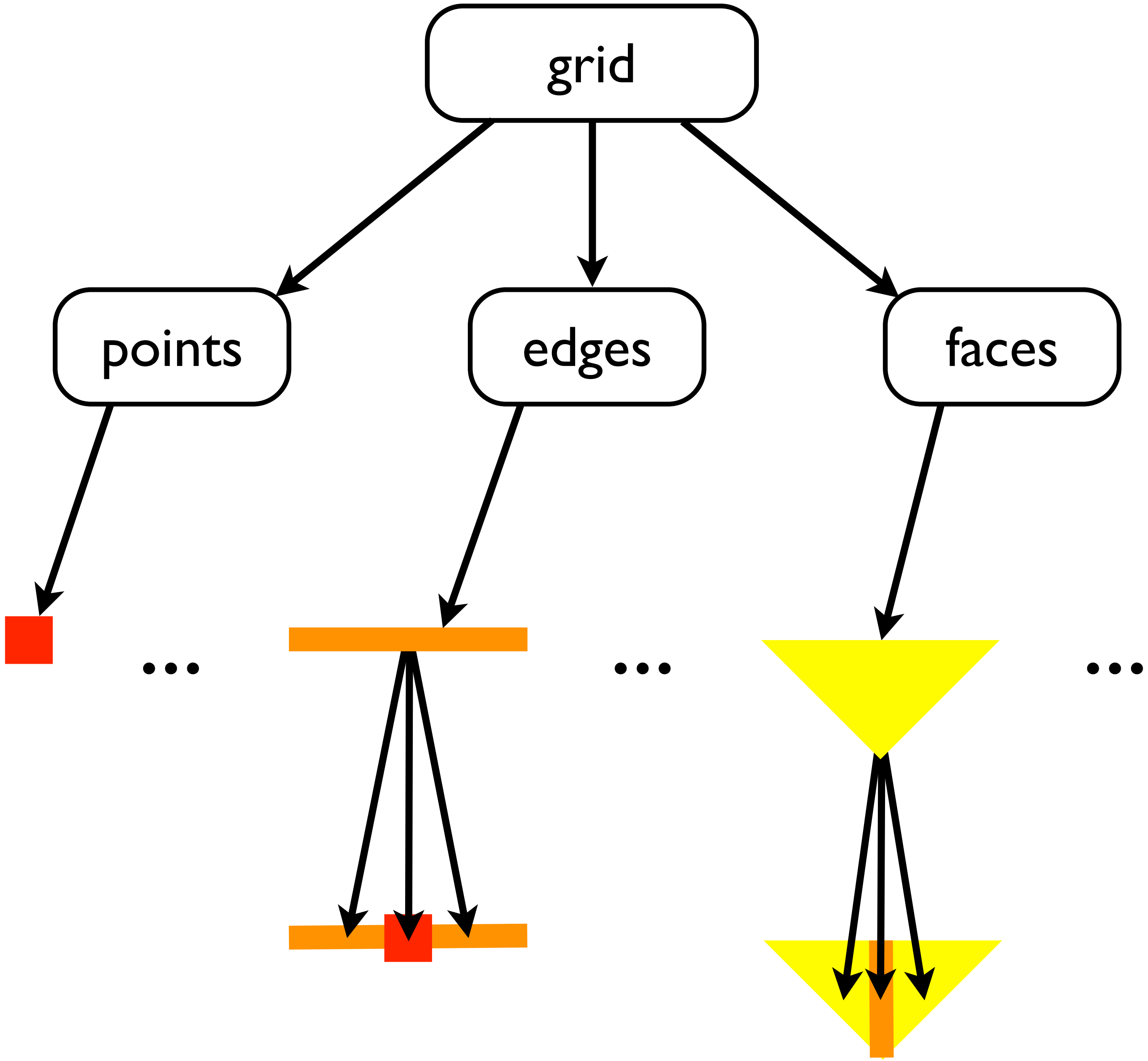
Adaptive grid structures: general “nd-tree”

- Subclasses in Matlab: “Quadtree < Ndtree”, “Octree < Ndtree”
- Grid is kept *graded*, via recursive refinement of neighbors.
- For visualization and computation of level sets: Delaunay triangulation/ tetrahedralization

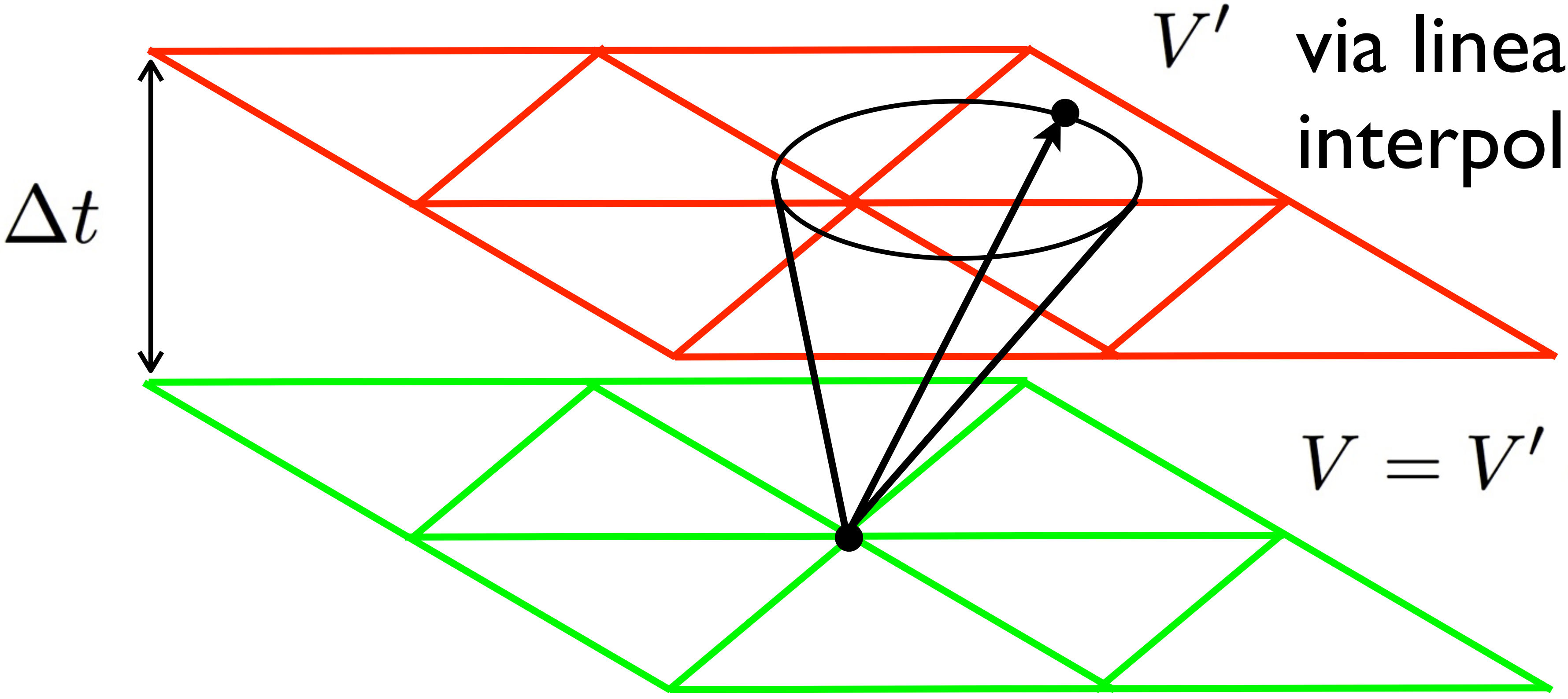


Adaptive grid structures: triangles

- Grid refinement initiated by edge refinement.
- Edge refinement based on error estimate.
- Grid kept *graded* by simply refining hypoteneuse first.
- Fast binary triangle search

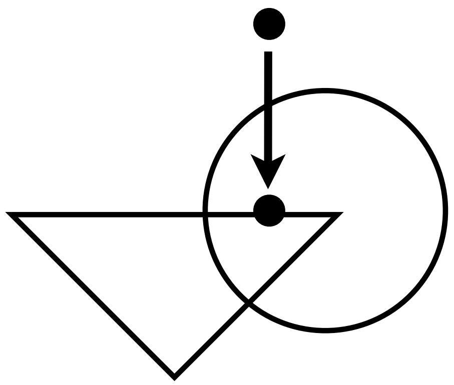


Godunov scheme



V' via linear triangle interpolation

$$V = V' + W|\mathbf{u}|^2 \Delta t$$



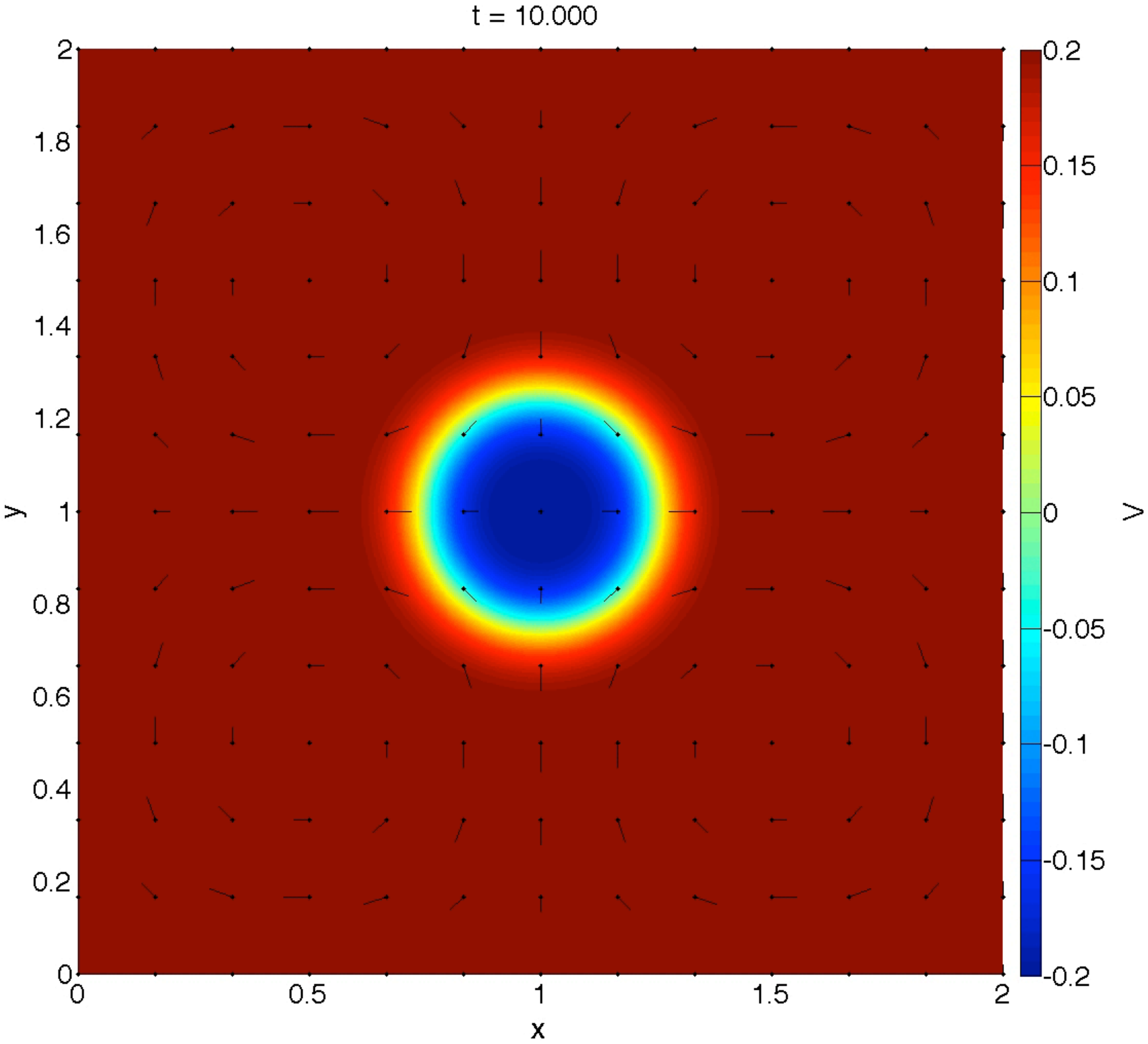
For each reachable triangle: contrained quadratic minimization, exact solution

Ex: 4-gyre flow (W = 10, t_f = 10, dt = 0.1)

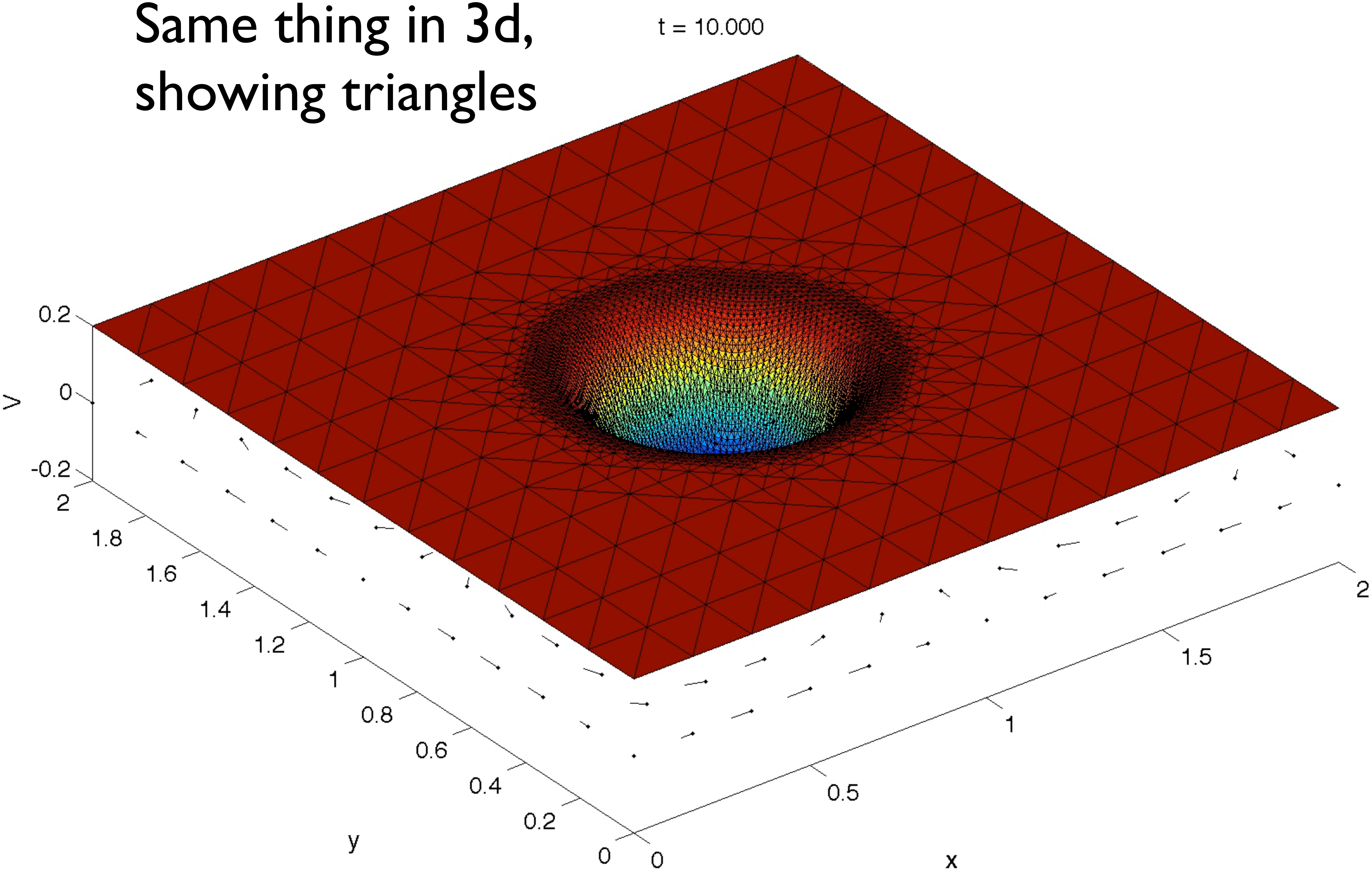
Animation of cost–to–go V and resulting trajectories, computed via Godunov scheme on adaptive triangle grid

Ex: 4-gyre flow ($W = 10$, $t_f = 10$, $dt = 0.1$)

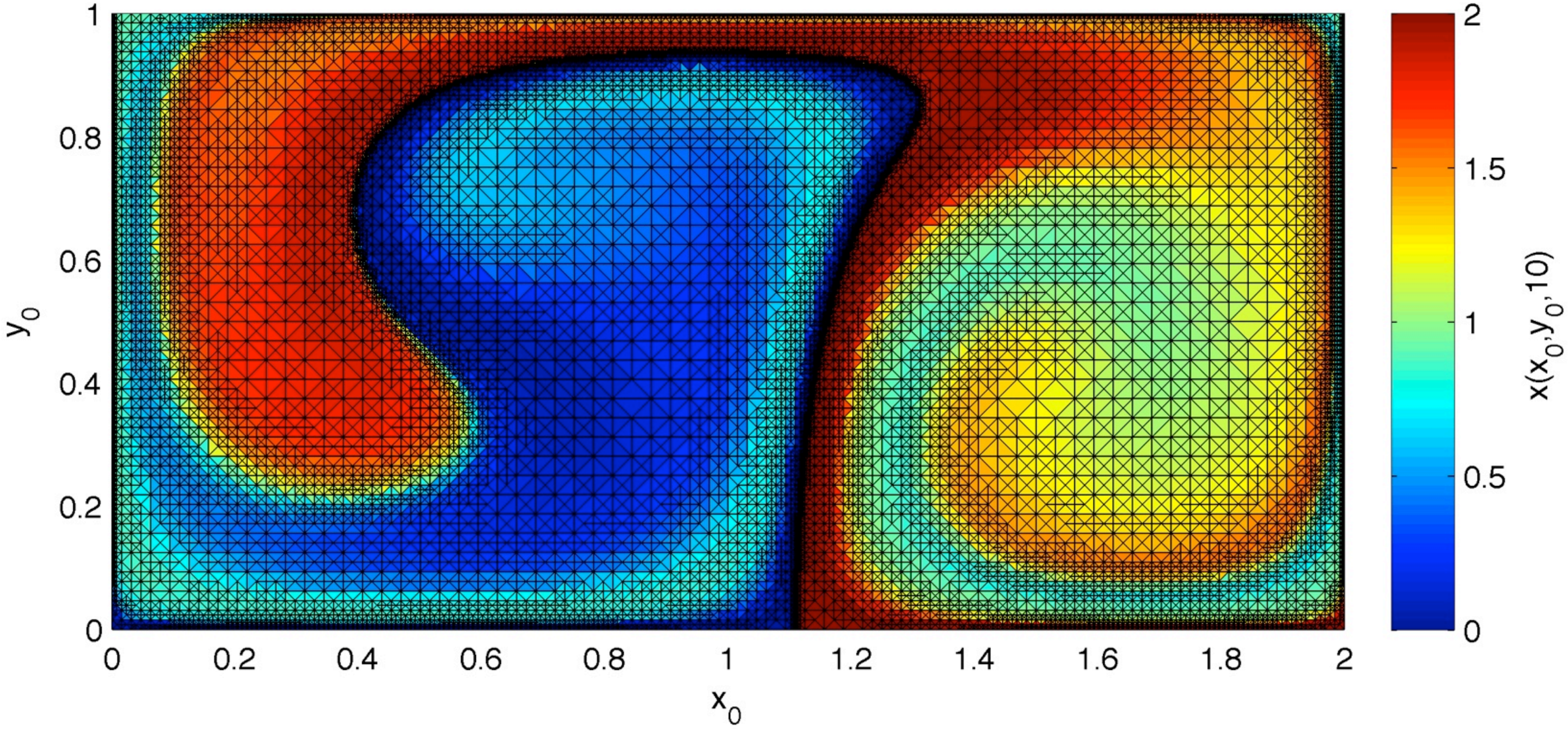
Animation of cost–to–go V and resulting trajectories, computed via Godunov scheme on adaptive triangle grid



Same thing in 3d, showing triangles

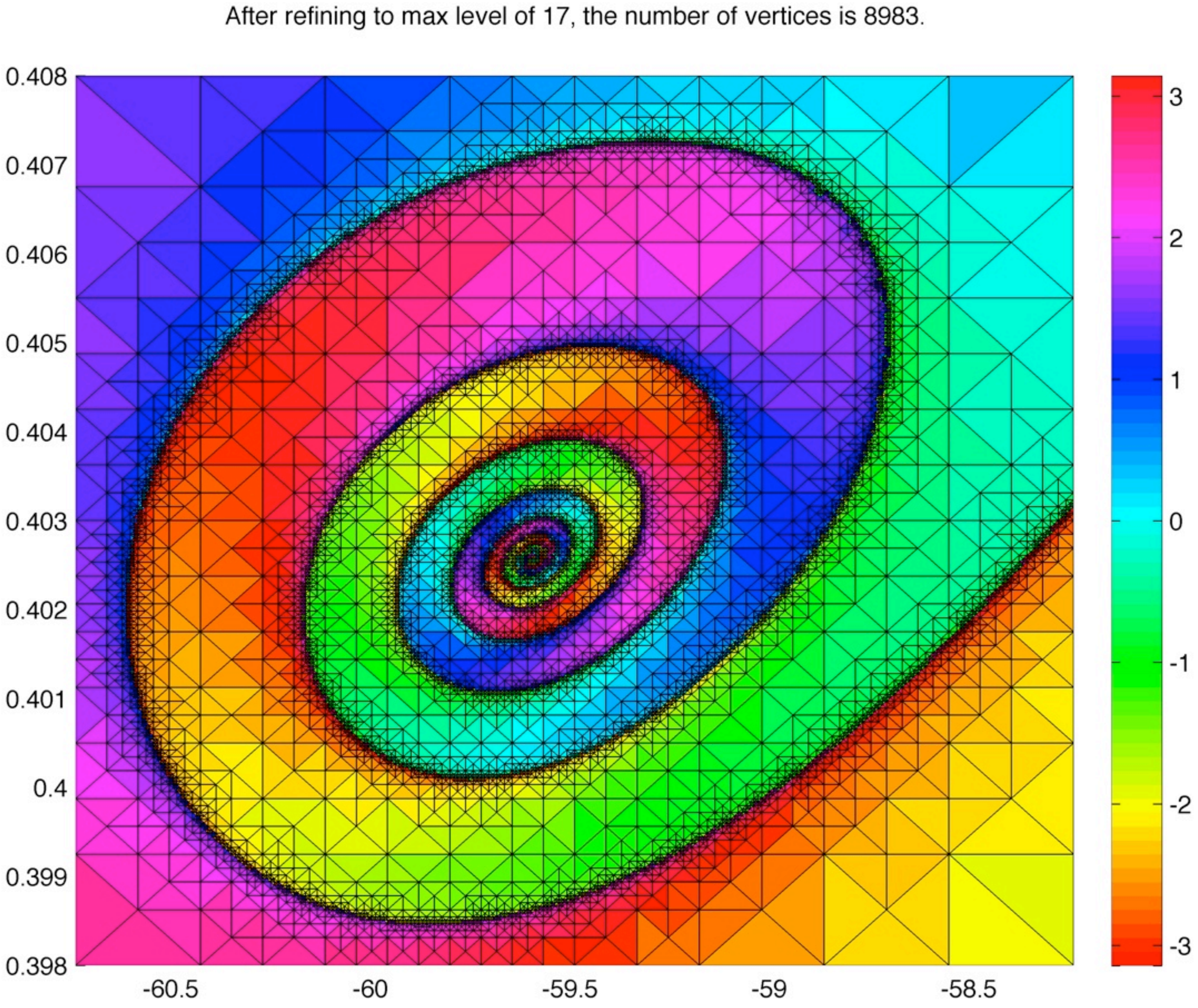


Related application: Structure of the flow field itself

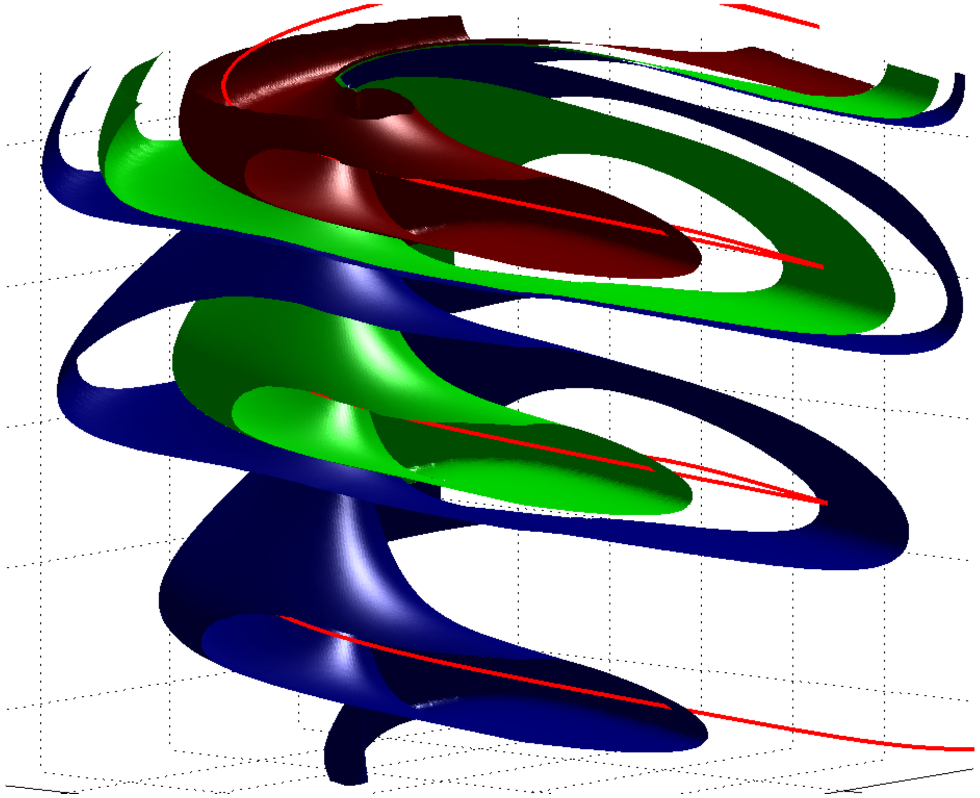


Future x-coordinate of passive Lagrangian particles in time-varying “2-gyre” flow, via adaptive triangular mesh

Another application: neuron phase dynamics



Phase function near unstable fixed point of 2D Hodgkin–Huxley neuron model, via **quadtree (nd–tree)**



3 Isochrons (level sets of phase function) near limit cycle (red) of Hindmarsh – Rose 3D bursting neuron model, via **octree (nd–tree)** adaptive mesh, **Delaunay tetrahedralization**, and custom **marching tetrahedra** isosurface function

Conclusion

- Demonstrated suitability of *particle* and *Godunov* methods for *minimum time* problem and *minimum energy* problem, respectively.
- Discovered importance of *remeshing* and *trimming* in Lagrangian method, and the power of a simple *adaptive triangle grid* in semi-Lagrangian method.
- Developed *"nd-tree" adaptive grid* for visualization of phase dynamics in 2D and 3D neuron models.
- **Impact**: Naval Research Lab has expressed interest in testing forward minimum time algorithm (published 2013) on operational gliders during experiments scheduled for July.

Summary of coding projects

Year	Location	Description	Method(s)	Pub-lished?	C++	Matlab
2013	UCSB	Min energy / end cost	Godunov semi-Lagrangian, adaptive triangle mesh	in prep-aration	2000	plots
2013	UCSB	Neuron phase dynamics	quadtree/octree (nd-tree)	in prep-aration	0	2000
2011	United Technologies	Obstacle/enemy avoidance	FM w/ speed optimization	no	7000	plots etc
2011	United Technologies	Multi-objective optimal control	augmented dynamic programming, PRM	no	1000	plots
2011	UCSB	Min time (without feedback)	1D front tracking w/ remeshing & trimming	yes	0	~10000
2009	UCSB	Min time (with feedback)	2D front tracking w/ remeshing	yes	0	~10000
2009	Sandia Nat'l Lab	Combustion reaction simulation	CSP w/ tabulation, kd-trees in ANN library	yes	small mods	plots

Thank you.

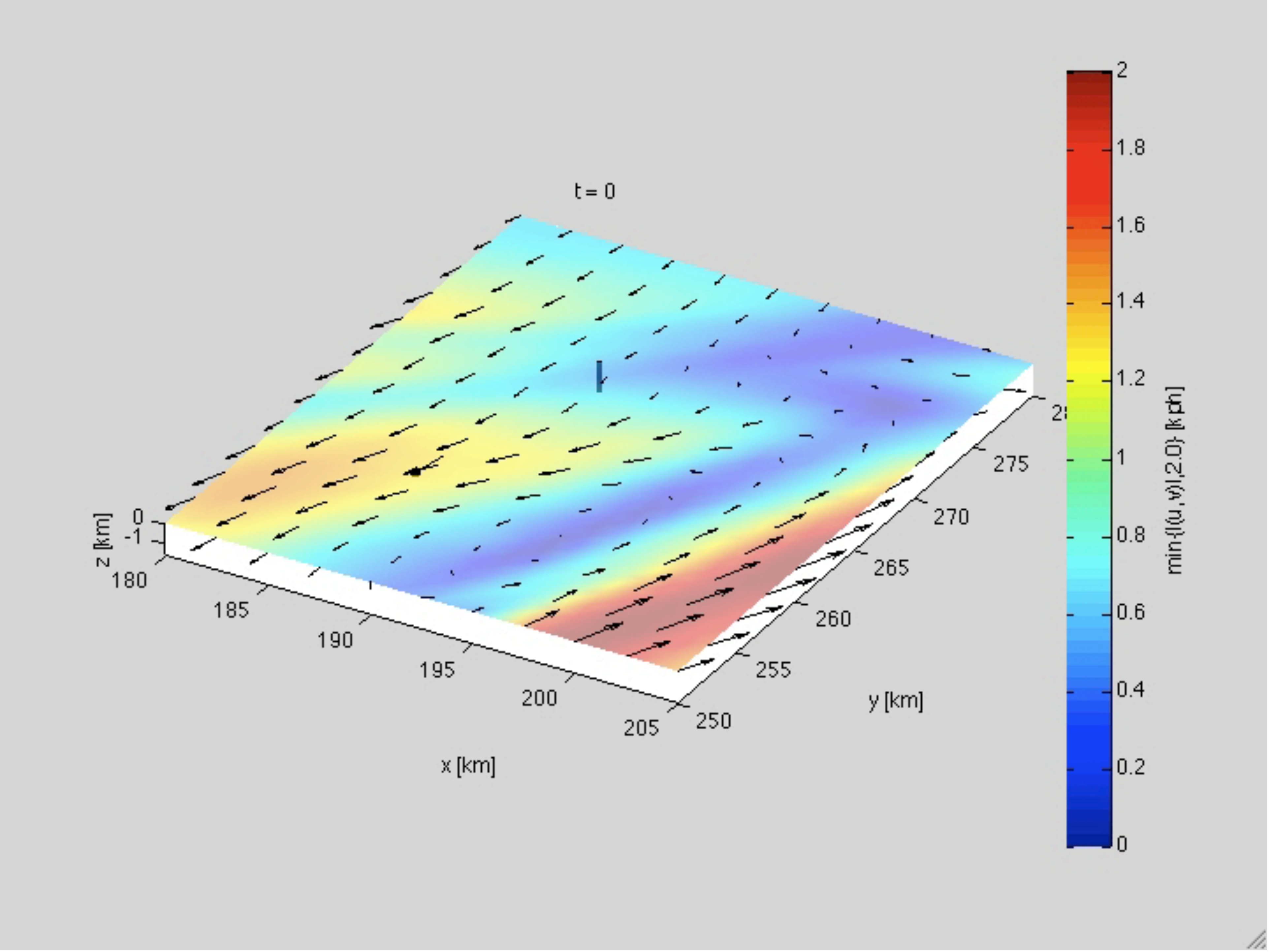
Acknowledgements:

- Office of Naval Research
- Igor Mezic, UCSB (advisor)
- Drew Poje, CUNY CSI (collaborator)
- Alex Mauroy, UCSB (collaborator), Frederic Gibou, UCSB (Level Set Methods), Alex Vladimirsky, Cornell

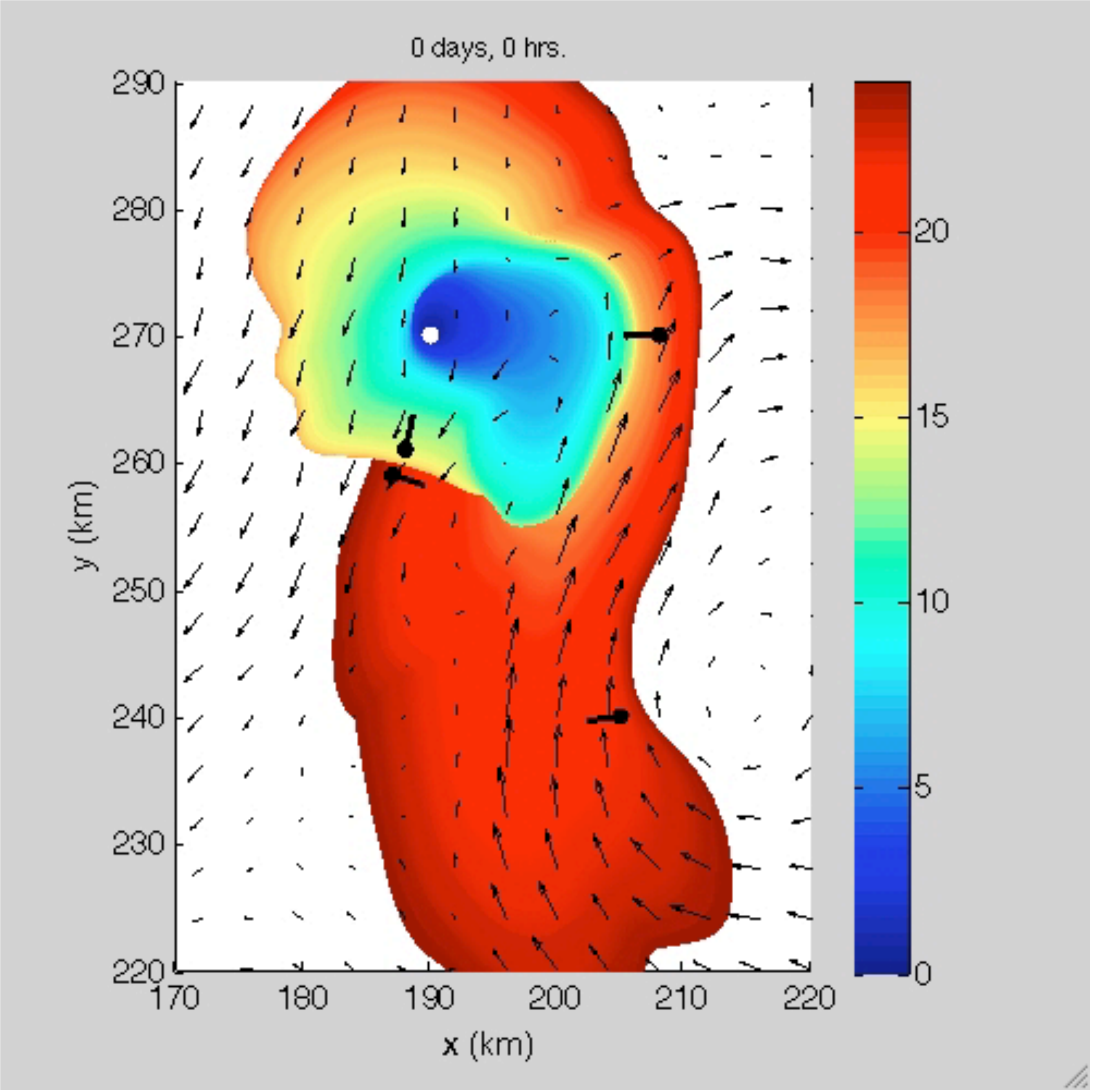
Publications:

- Mauroy, Alexandre, Blane Rhoads, Igor Mezić, and Jeff Moehlis. "Global Isochrons and Phase Sensitivity of Bursting Neurons." SIAM Journal of Applied Dynamical Systems (IN PREPARATION).
- Rhoads, Blane, Igor Mezić, and Andrew C. Poje. "Minimum time heading control of underpowered vehicles in time-varying ocean currents." Ocean Engineering 66 (2013): 12-31.
- Debusschere, Bert J., et al. "Computational singular perturbation with non-parametric tabulation of slow manifolds for time integration of stiff chemical kinetics." Combustion Theory and Modelling 16.1 (2012): 173-198.
- Rhoads, Blane, Igor Mezić, and Andrew Poje. "Minimum time feedback control of autonomous underwater vehicles." Decision and Control (CDC), 2010 49th IEEE Conference on. IEEE, 2010.
- Radloff, Stefan, et al. "First wafer delay and setup: How to measure, define and improve first wafer delays and setup times in semiconductor fabs." Advanced Semiconductor Manufacturing Conference, 2009. ASMC'09. IEEE/SEMI. IEEE, 2009.

3d, time-varying flow

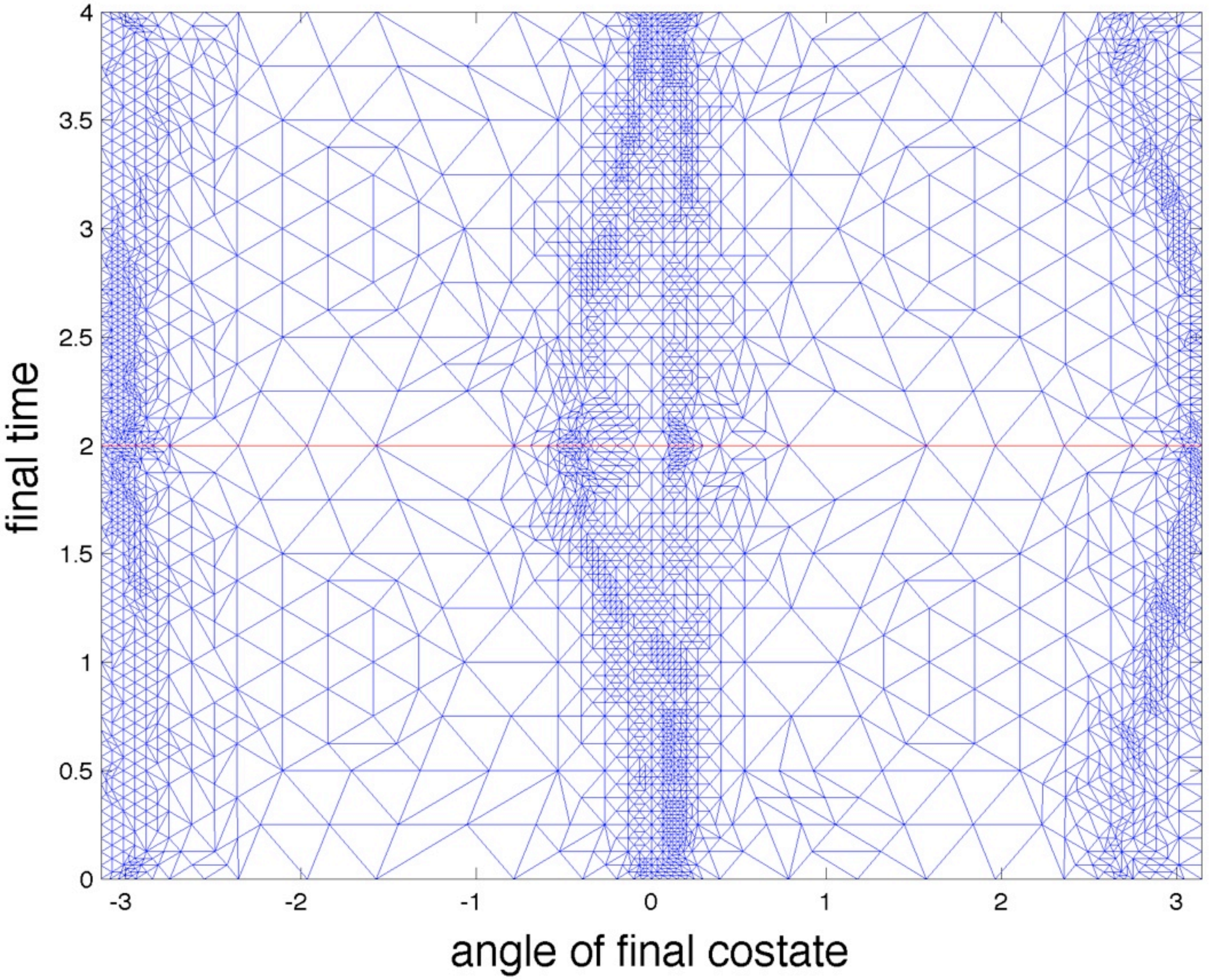


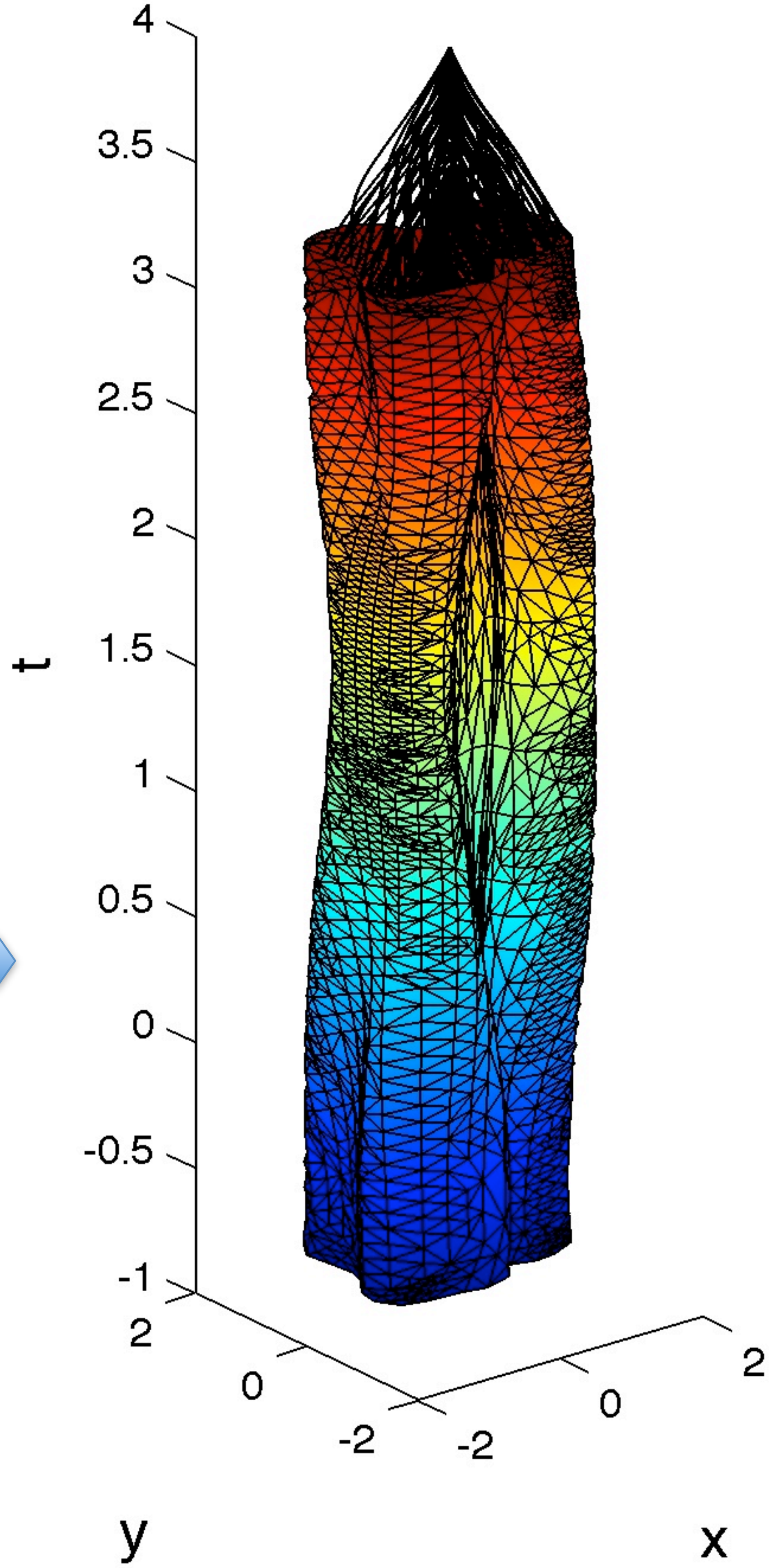
Min time via backward particle method



From [Rhoads, et. al (2010)]

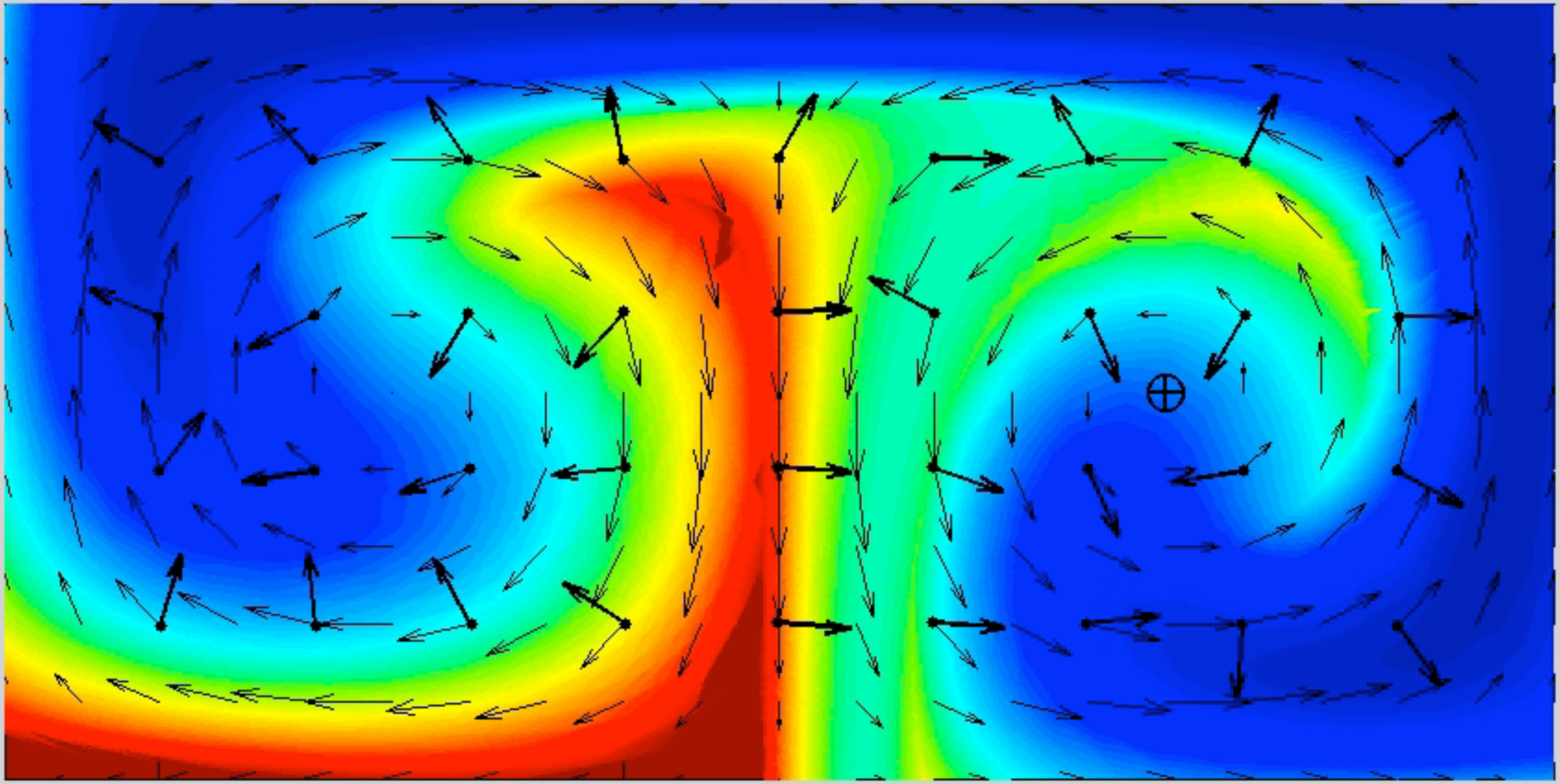
Grid used



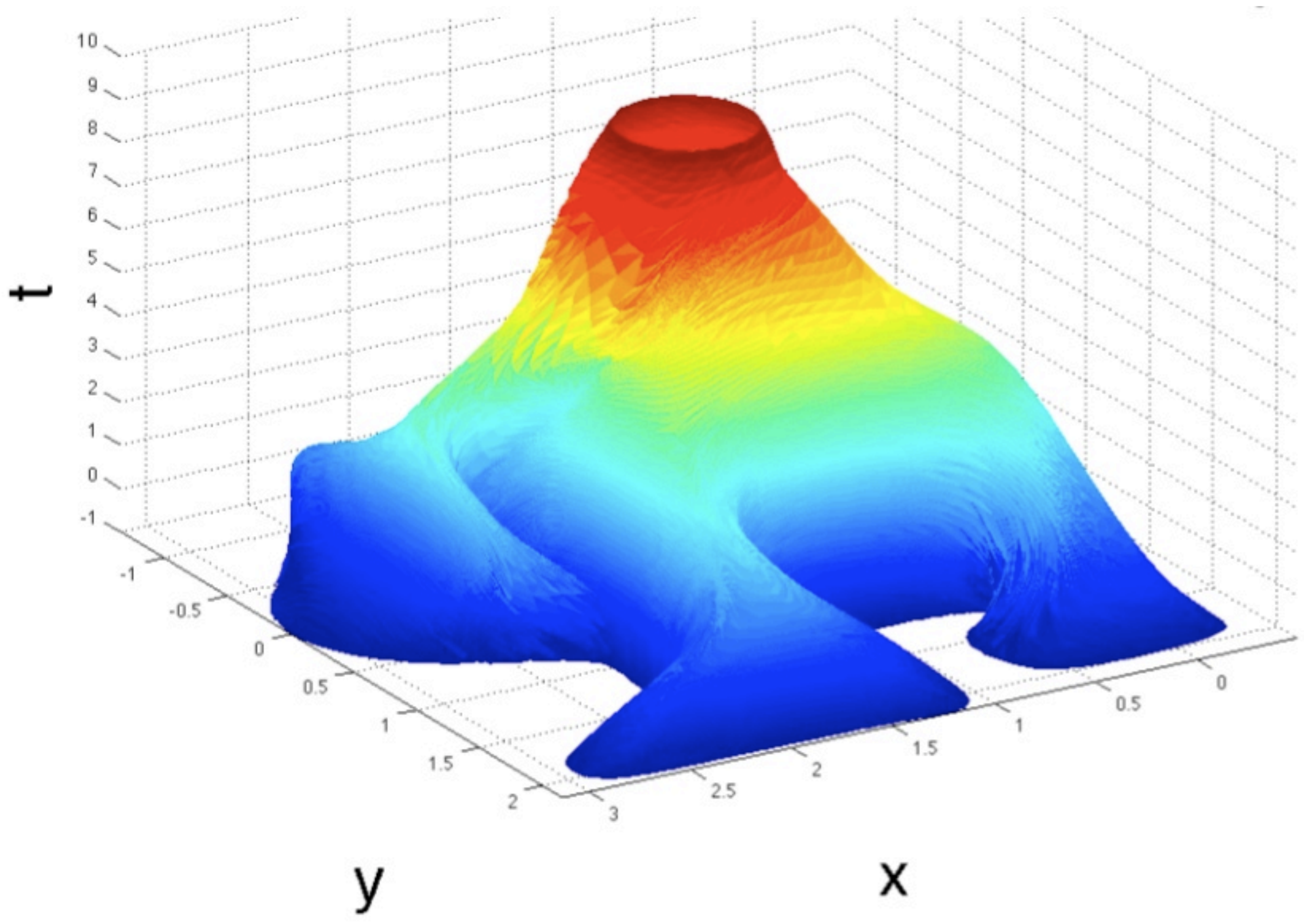


Extremals and final controllability front T(**x**,t) = T_{max}

Min energy / end cost via backward Lagrangian method

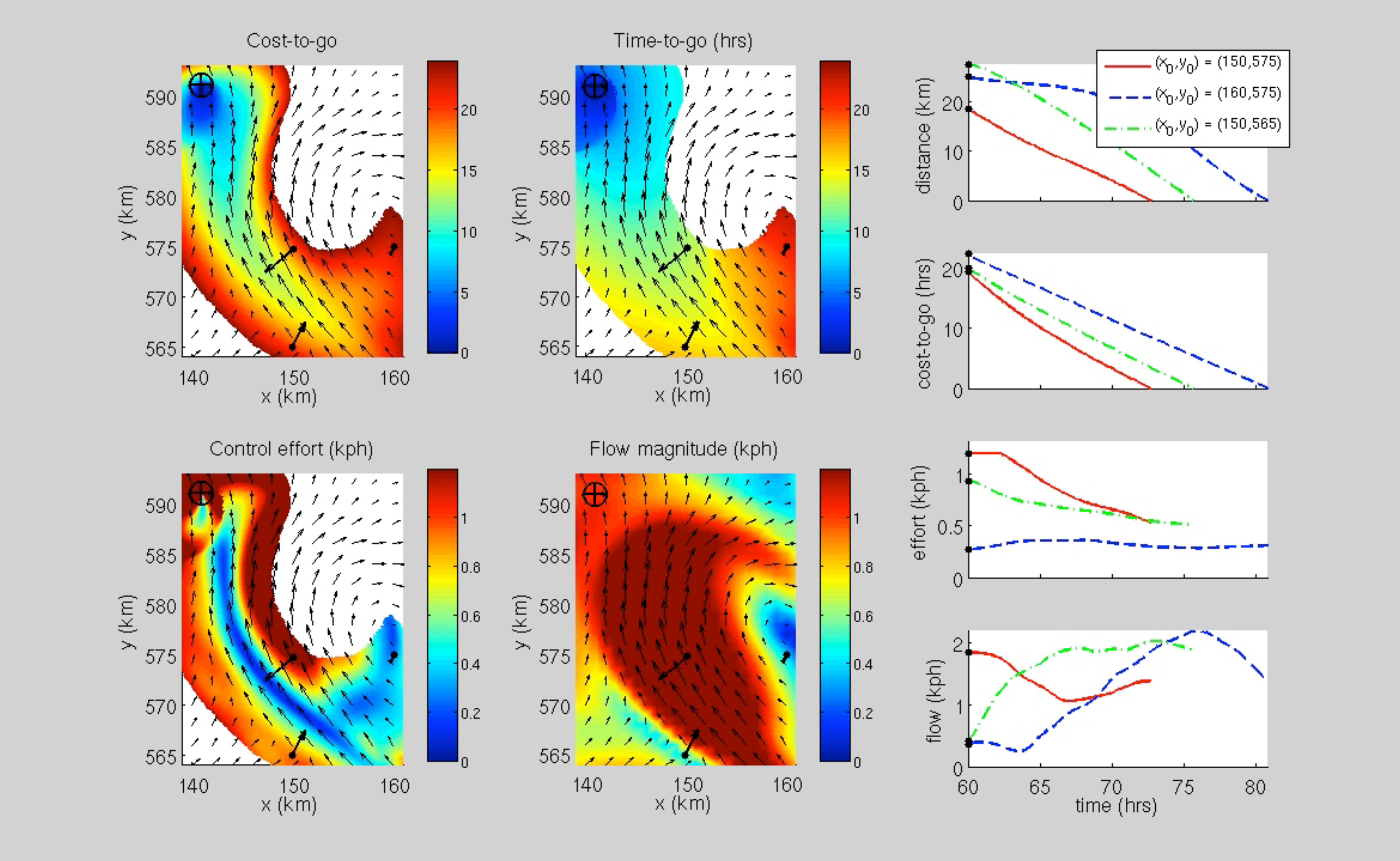


Min energy / end cost 2D tracked front



Level set of V (truly 2D front in 3D state space) for min energy (variable vehicle speed) problem, computed via particle method and (preliminary) adaptive triangular mesh.

Min time/energy - particle method



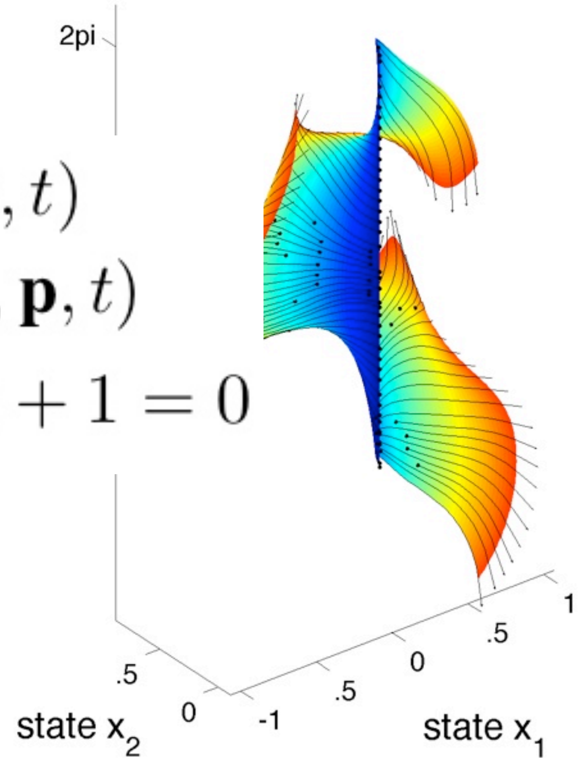
How is this done?
“Euler Lagrange backwards integration”

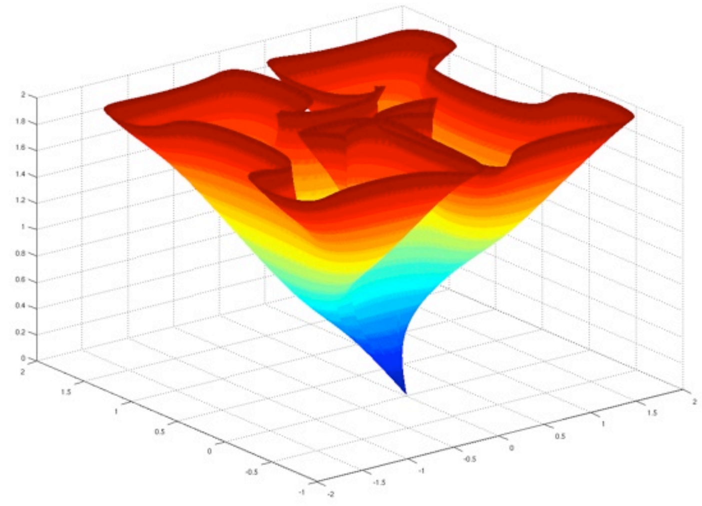
$$\dot{\mathbf{x}} = \mathbf{v}(\mathbf{x},t) - s\frac{\mathbf{p}}{|\mathbf{p}|} \quad = H^*_{\mathbf{x}}(\mathbf{x},\mathbf{p},t)$$

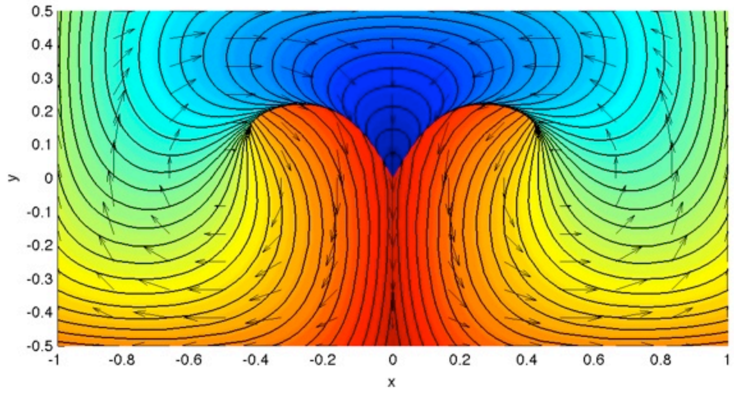
$$\dot{\mathbf{p}} = -\nabla\mathbf{v}(\mathbf{x},t)^T\mathbf{p} \quad = -H^*_{\mathbf{p}}(\mathbf{x},\mathbf{p},t)$$

$$\left[\mathbf{v}(\mathbf{x}(t_f),t_f) - s\frac{\mathbf{p}(t_f)}{|\mathbf{p}(t_f)|}\right]\cdot\mathbf{p}(t_f)+1=0$$

$$\mathbf{x}(t_f)=\mathbf{x}_f$$







$$\begin{tikzcd}
\text{Hamiltonian formulation} & (x,p)\text{-space} \arrow[r, "\text{unique}"] & \text{smooth } W^s(0,0) \arrow[d, "\text{project}"] \\
\text{Dynamic programming} & (x,u)\text{-space} \arrow[u, "\text{lift}"] \arrow[r, "\text{non-unique}"] & \text{feedback } (x,t)\mapsto u^*
\end{tikzcd}$$

[Osinga, Hauser (2005)]

Going forward

costate equation

$$\dot{p} = -v_x p - w_x q$$
$$\dot{q} = -v_y p - w_y q$$

end condition:

$$|\mathbf{p}| = \frac{1}{s - \mathbf{v} \cdot \frac{\mathbf{p}}{|\mathbf{p}|}}$$

Tranform to polar

$$(p, q) = -r(\cos\theta, \sin\theta)$$

~~$\dot{r} = r\left\{v_x \cos^2\theta + (v_y + w_x)\sin\theta\cos\theta + w_y \sin^2\theta\right\}$~~

$$\dot{\theta} = -v_y \cos^2\theta + (v_x + w_y)\sin\theta\cos\theta + w_x \sin^2\theta$$

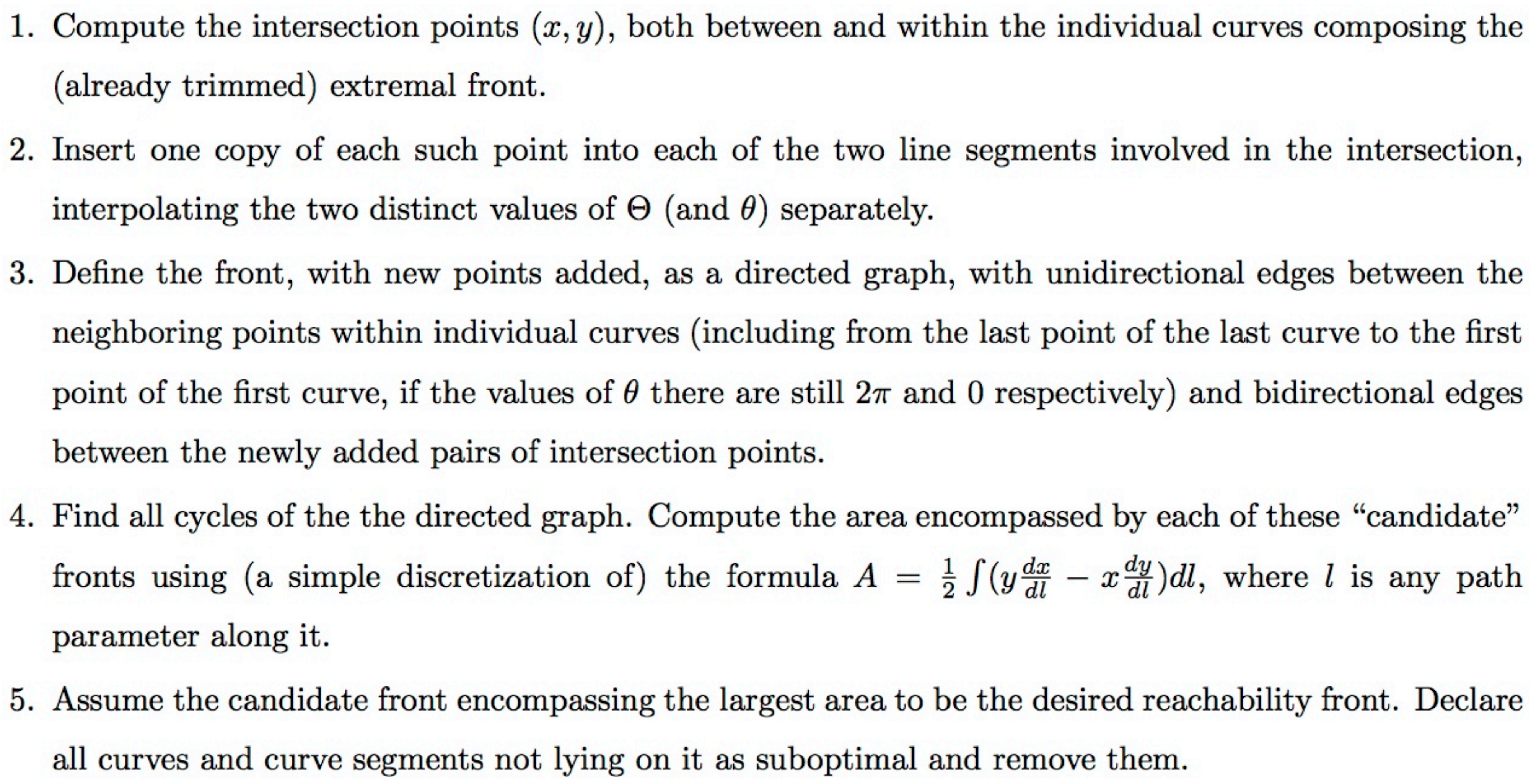
$$r = \frac{1}{s - v\cos\theta + w\sin\theta}$$

Along a minimum time trajectory,
magnitude of costate can be scaled as desired!

Reduced minimum time Euler Lagrange equations & end condition:

$$\dot{x} = v(x, y, t) + s\cos\theta$$
$$\dot{y} = w(x, y, t) + s\sin\theta$$
$$\dot{\theta} = -v_y \cos^2\theta + (v_x + w_y)\sin\theta\cos\theta + w_x \sin^2\theta$$

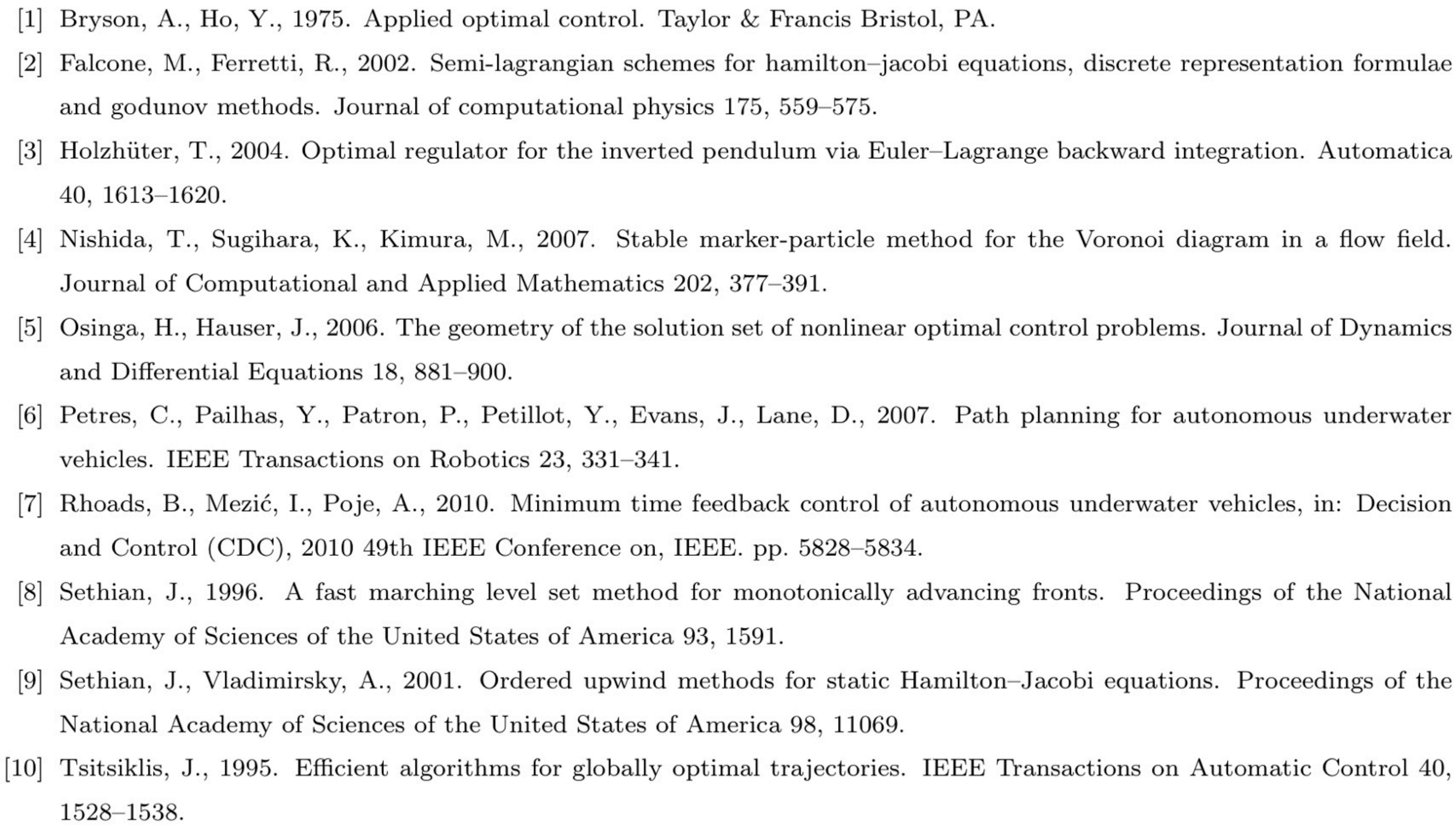
Trimming procedure:



1. Compute the intersection points (x, y), both between and within the individual curves composing the (already trimmed) extremal front.
2. Insert one copy of each such point into each of the two line segments involved in the intersection, interpolating the two distinct values of Θ (and θ) separately.
3. Define the front, with new points added, as a directed graph, with unidirectional edges between the neighboring points within individual curves (including from the last point of the last curve to the first point of the first curve, if the values of θ there are still 2π and 0 respectively) and bidirectional edges between the newly added pairs of intersection points.
4. Find all cycles of the the directed graph. Compute the area encompassed by each of these "candidate" fronts using (a simple discretization of) the formula $A = \frac{1}{2}\int(y\frac{dx}{dl} - x\frac{dy}{dl})dl$, where l is any path parameter along it.
5. Assume the candidate front encompassing the largest area to be the desired reachability front. Declare all curves and curve segments not lying on it as suboptimal and remove them.

From [Rhoads, et. al (2013)]

Other references:



[1] Bryson, A., Ho, Y., 1975. Applied optimal control. Taylor & Francis Bristol, PA.

[2] Falcone, M., Ferretti, R., 2002. Semi-lagrangian schemes for hamilton–jacobi equations, discrete representation formulae and godunov methods. Journal of computational physics 175, 559–575.

[3] Holzhüter, T., 2004. Optimal regulator for the inverted pendulum via Euler–Lagrange backward integration. Automatica 40, 1613–1620.

[4] Nishida, T., Sugihara, K., Kimura, M., 2007. Stable marker-particle method for the Voronoi diagram in a flow field. Journal of Computational and Applied Mathematics 202, 377–391.

[5] Osinga, H., Hauser, J., 2006. The geometry of the solution set of nonlinear optimal control problems. Journal of Dynamics and Differential Equations 18, 881–900.

[6] Petres, C., Pailhas, Y., Patron, P., Petillot, Y., Evans, J., Lane, D., 2007. Path planning for autonomous underwater vehicles. IEEE Transactions on Robotics 23, 331–341.

[7] Rhoads, B., Mezić, I., Poje, A., 2010. Minimum time feedback control of autonomous underwater vehicles, in: Decision and Control (CDC), 2010 49th IEEE Conference on, IEEE. pp. 5828–5834.

[8] Sethian, J., 1996. A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Sciences of the United States of America 93, 1591.

[9] Sethian, J., Vladimirsky, A., 2001. Ordered upwind methods for static Hamilton–Jacobi equations. Proceedings of the National Academy of Sciences of the United States of America 98, 11069.

[10] Tsitsiklis, J., 1995. Efficient algorithms for globally optimal trajectories. IEEE Transactions on Automatic Control 40, 1528–1538.