Centro Euro-Mediterraneo sui Cambiamenti Climatici

www.cmcc.it

An introduction to the VISIR-2 ship weather routing model

Gianandrea Mannarini

CMCC Foundation – Euro-Mediterranean Center on Climate Change

Institute for Earth System Modeling
Global Coastal Ocean Division
via Marco Biagi 5, 73100 Lecce (Italy)

latest update: 2024-08-05

VISIR-2 resources

open access - open review journal paper

https://doi.org/10.5194/gmd-17-4355-2024

Geosci. Model Dev., 17, 4355–4382, 2024
https://doi.org/10.5194/gmd-17-4355-2024

Geoscientific Model Development
Open Access
EGU

Model description paper

VISIR-2: ship weather routing in Python

Gianandrea Mannarini[1], Mario Leonardo Salinas[1], Lorenzo Carelli[1], Nicola Petacco[2], and Josip Orović[3]

[1]CMCC Foundation – Euro-Mediterranean Center on Climate Change, Lecce, Italy
[2]DITEN, Università degli Studi di Genova, via Montallegro 1, 16145 Genoa, Italy
[3]Maritime Department, University of Zadar, Ul. Mihovila Pavlinovića, 23000 Zadar, Croatia

open language - open source model code

https://zenodo.org/communities/visir2

April 11, 2024 (v3) | Software | Open

[VISIR-2 ship weather routing model] source code (Python)

Salinas, Mario Leonardo; Carelli, Lorenzo; Mannarini, Gianandrea

Source code of the VISIR-2 ship weather routing model. The Python-refactored VISIR-2 model considers currents, waves, and wind to optimise routes. The model was validated, and its computational performance is quasi-linear. For a ferry sailing in the Mediterranean Sea, VISIR-2 yields the largest percentage emission savings for upwind navigation...

Part of VISIR-2 ship weather routing model (Python)

Uploaded on April 11, 2024

2 more versions exist for this record

Outline

1. *Introduction*
2. *Model workflow*
3. *Case studies*
4. *Discussion*

Outline

1. *Introduction*
2. *Model workflow*
3. *Case studies*
4. *Discussion*

Why an open-source ship weather routing model?

- *Reducing costs and emissions from shipping is crucial for both the economy and the environment*

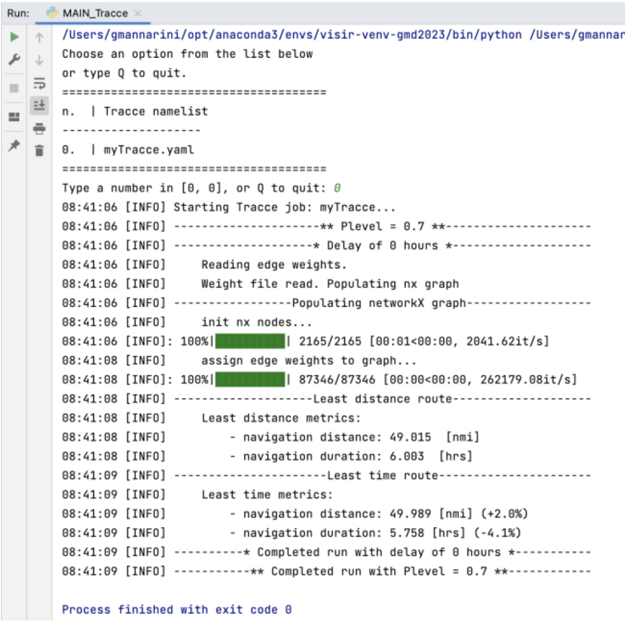
- *weather routing is a an operational tool to achieve this goal*

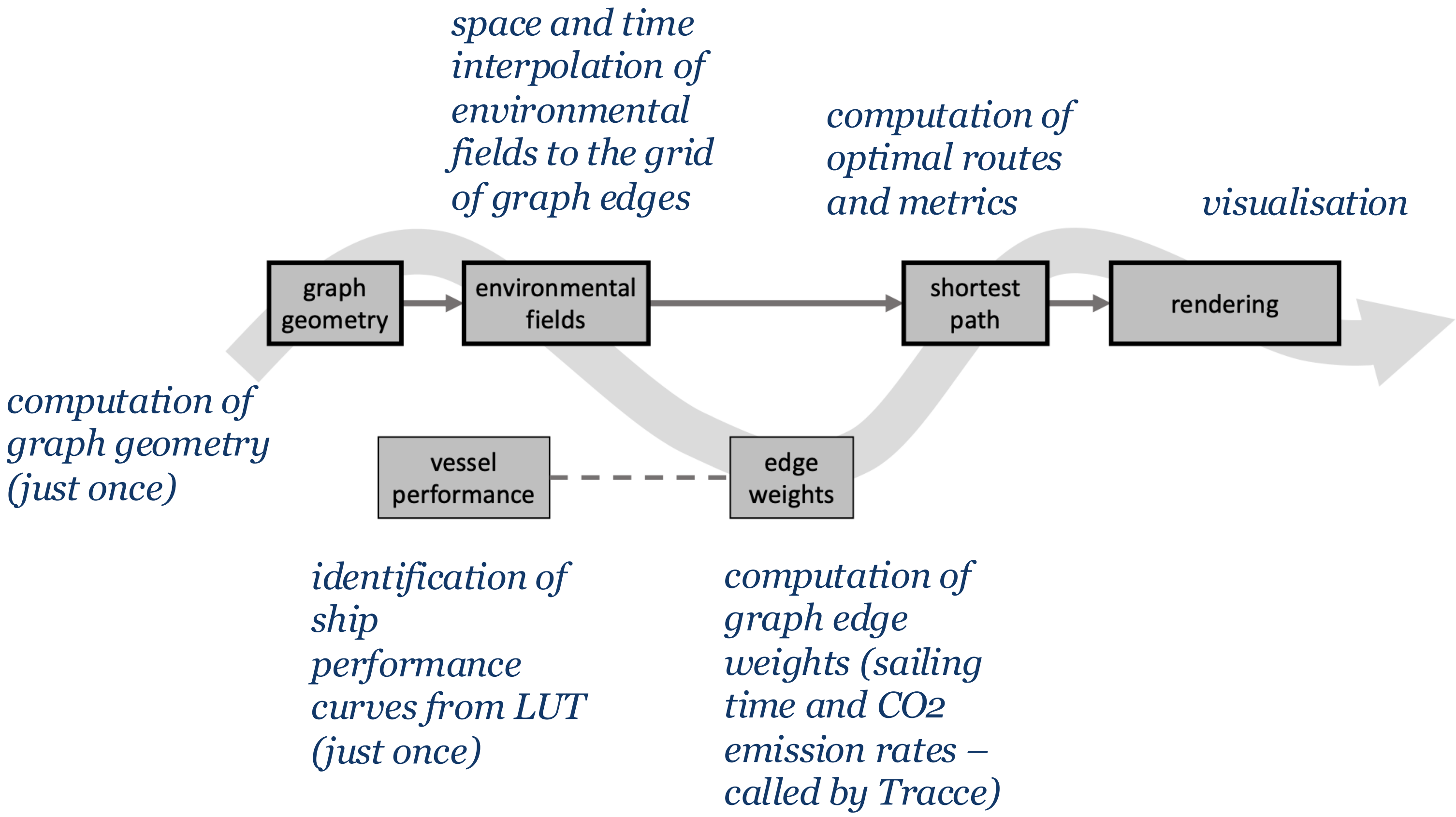
- *open-source models can achieve superior performance, foster cross-disciplinary developments, provide a public good for science and technology*

Outline

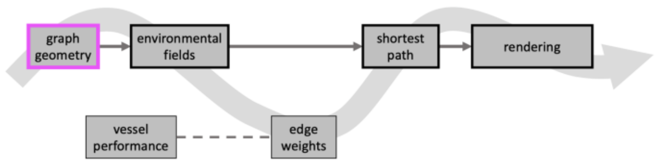
VISIR-2 workflow

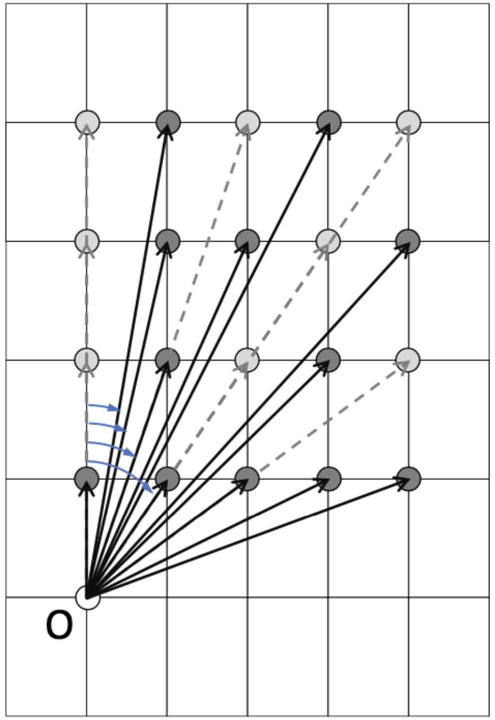
- *Greater modularity with respect to VISIR-1*
- *conda virtual environment ("visir-venv") for portability*





Graph structure





Graph stencil for connectivity

up to 4th order neighbours (ν=4)

Graph nodes are joined through edges, up to ν hops

the higher the ν, the smaller the minimum resolved angle and the smoother the ship's route

in VISIR-2, multi-hop collinear edges (dashed) are pruned.

Benefits:

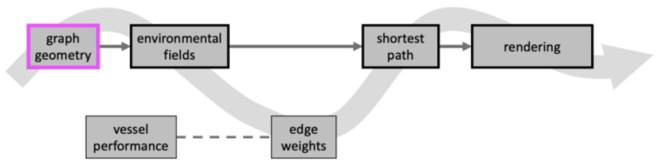
- *saving RAM memory*
- *more faithful representation of the environmental fields*

Mercator projection of the graph grid (`pyproj` library) used

ν	$\nu(\nu+1)$	N_{q1}
1	2	2
2	6	4
3	12	8
4	20	12
5	30	20
6	42	24
7	56	36
8	72	44
9	90	56
10	110	64

- *$\nu(\nu+1)$: number of edges in the first quadrant*
- *N_{q1} : number of edges upon pruning of collinear ones*
- *N_{q1} affects the memory allocation*

Graph computation

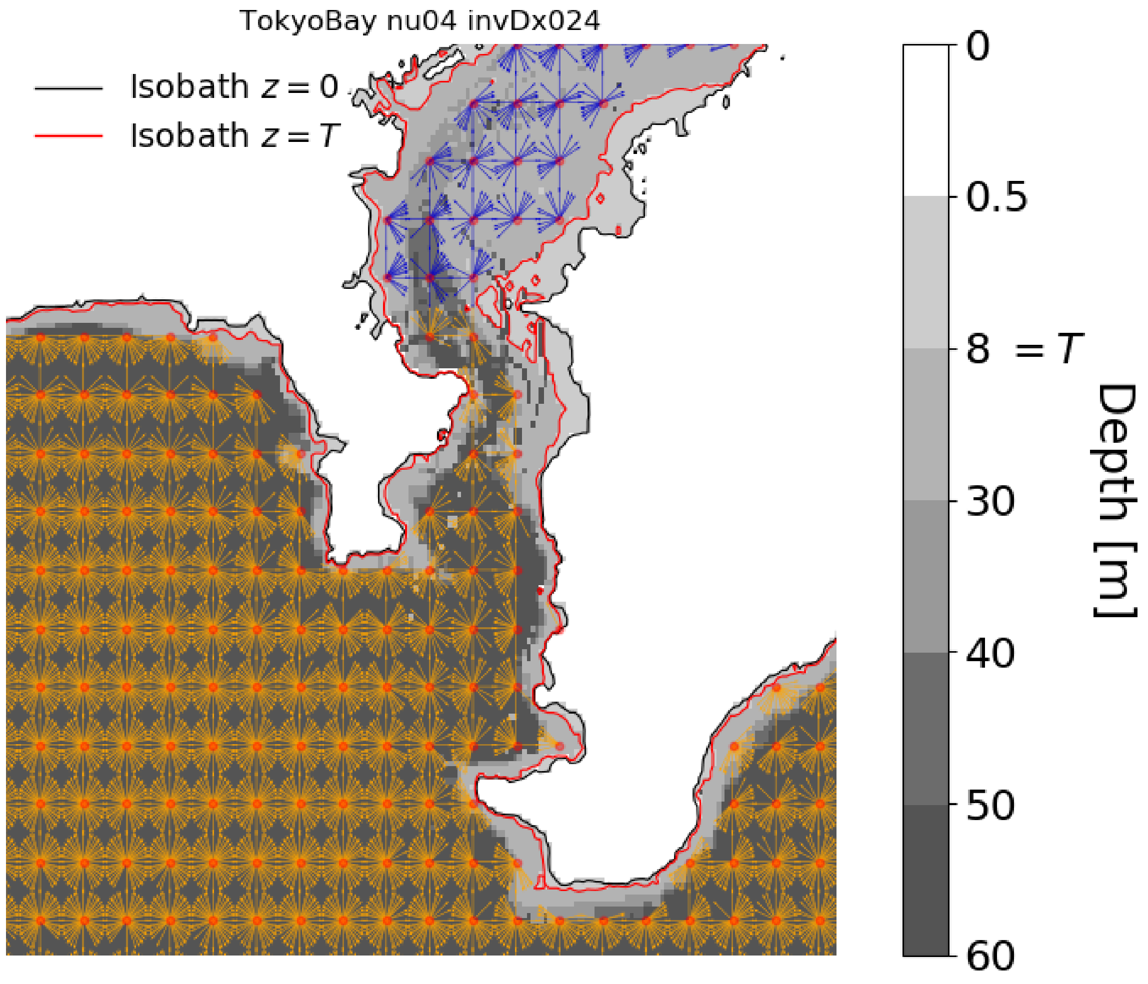


Graph edges indexed via a K-dimensional tree

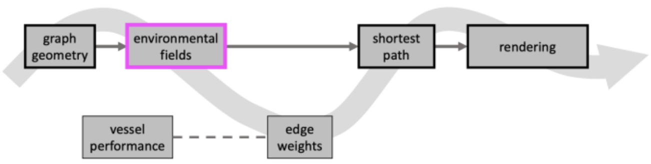
it can effectively be queried for:

- *nearest neighbours (coast proximity of nodes)*
- *range queries (coast intersection of edges)*

implementation in Python:
`scipy.spatial.KDTree`



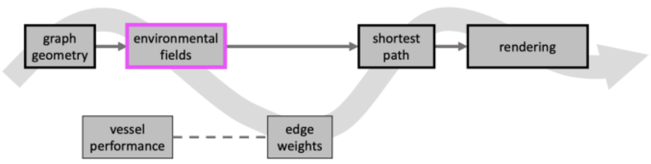
Input metocean data



dynamic environmental fields from data-assimilative models:

type	*product*	*Spatial resolution*	*Time resolution*
Waves	*MED-SEA_ANALYSISFORECAST_WAV_006_017*	*(1/24)º* *2.5 miles*	*1 hour*
Currents	*MEDSEA_ANALYSISFORECAST_PHY_006_013*	*(1/24)º* *2.5 miles*	*1 hour*
Wind	*Set I - HRES*	*(1/10)º* *6.0 miles*	*6 hours*

Sea over land

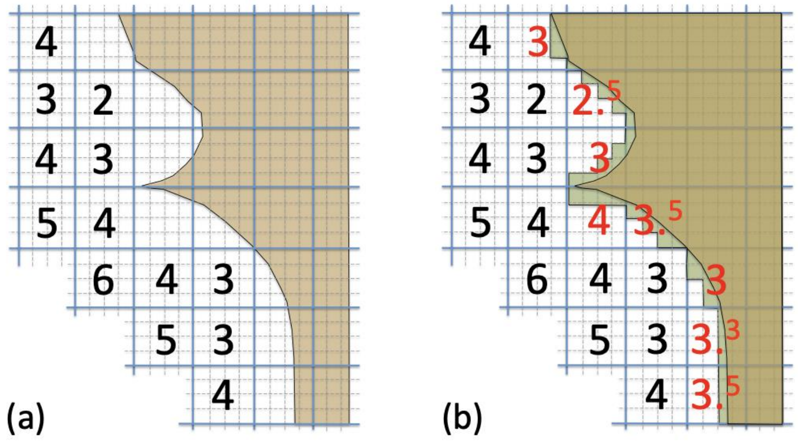


Limited spatial resolution of marine fields (waves, currents) leaves blank gridpoints in the vicinity of the physical shoreline

→ in VISIR-2, marine fields are extrapolated inshore via «sea over land»:

coastal missing values of sea fields are replaced with the average of the first neighbouring grid points

Note: sea over land is applied ahead of field interpolation (s. next slide)

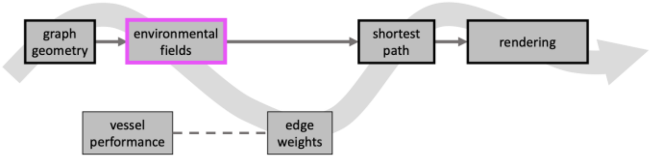


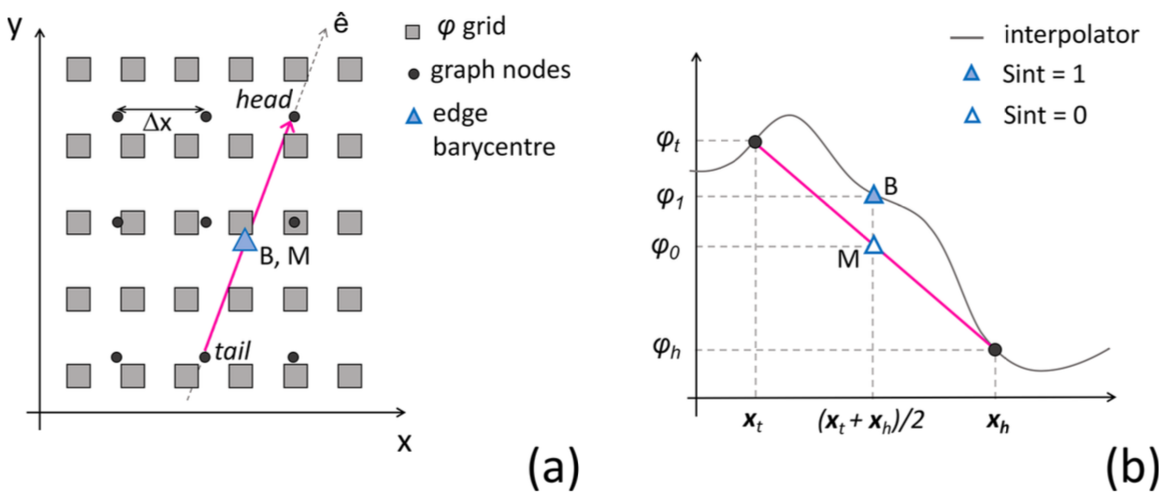
(a) Numbers represent original field values, with coastline (black line) and landmass (brown area)
(b) Field values after one sea-over-land iteration (replaced missing values are printed as red numbers). The land–sea mask of the target grid is shown in greenish colour

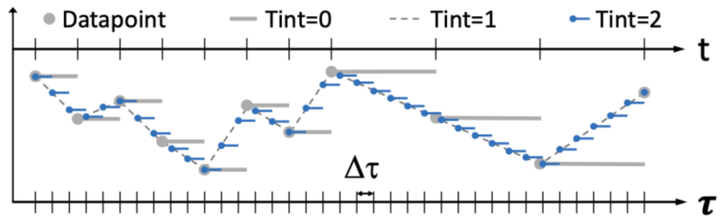
Mannarini et al 2016, https://doi.org/10.5194/gmd-9-1597-2016

cmcc

Spatial and temporal interpolation







Remap environmental fields to the graph grid:

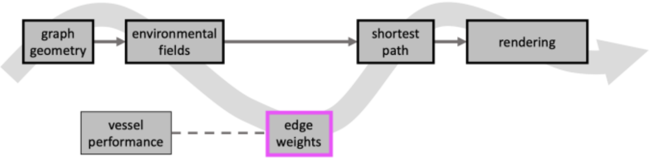
Two options:

- *averaging between the edge head and tail's values (`Sint = 0`, default)*
- *interpolating their values to the edge barycentre (`Sint = 1`)*

a) *environmental field values (grey dots) interpolated in time on a finer grid with Δτ spacing (`Tint = 2` or blue dots)*

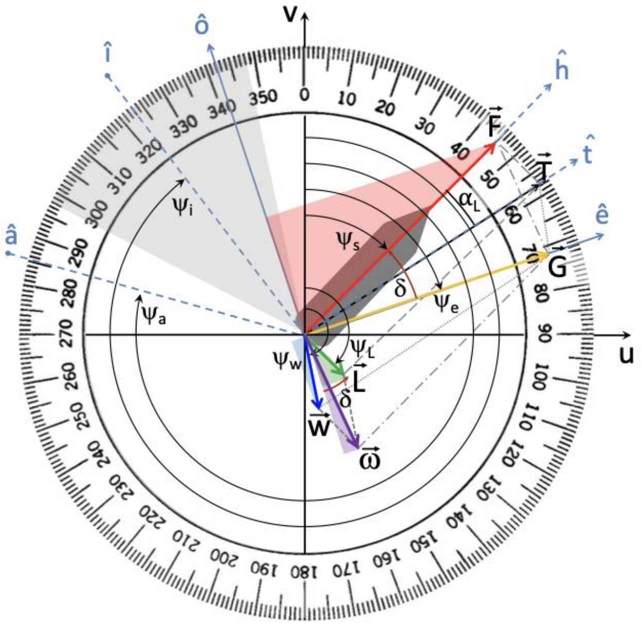
b) *edge weight at the nearest available timestep (floor function used, blue segments) is selected*

Kinematics



Key hypotheses:

- *linear superposition of velocities (speed through water, sea currents, leeway)*
- *ship's motion to occur along a graph edge*
- *speed through water resulting from sea state only (cf. «vessel performance curve»)*



F: forward speed (along vessel's heading)

T: speed through water (differs from F if leeway present)

G: speed over ground (along vessel's course – graph edge)

w: ocean current

L: leeway speed

→ *STW as a vector sum of F and leeway*

$$\vec{T} = \vec{F} + \vec{L}$$

→ *SOG as a vector sum of:*

STW and current (no leeway)

or:

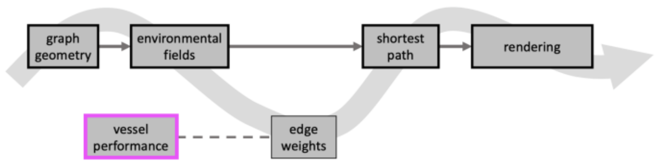
F and ω *(in general)*

$$SOG = \omega_{\parallel} + \sqrt{F^2 - \omega_{\perp}^2}$$

→ *angle of attack δ between ship's heading and course for offsetting cross-flow*

$$\delta = \psi_h - \psi_e$$

Vessel performance curve: ferry

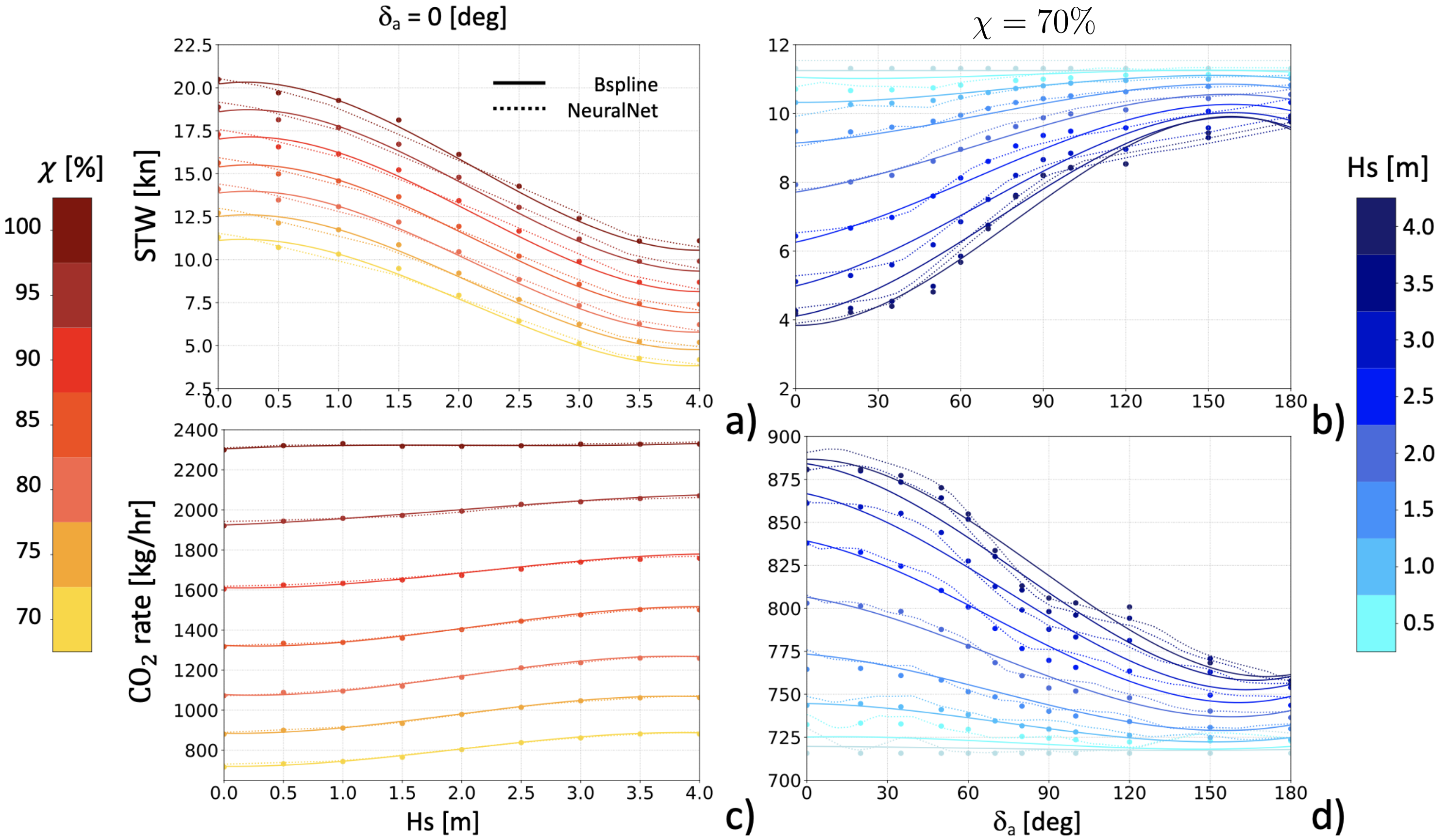


use of University of Zadar ship command-bridge/ engine-room coupled simulator:

- *wind waves*
- *no leeway*
- *explored dependence of STW on*
  - *engine load (χ)*
  - *significant wave height (H_s)*
  - *relative wave direction (δ_a)*
- *outcome interpolated via a neural network (multi-layer perceptron)*

Interreg Italy - Croatia GUTTA

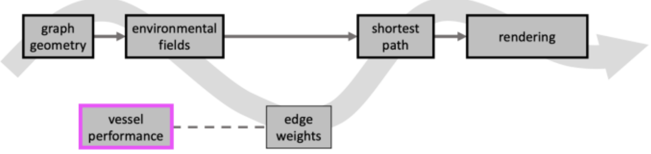
EUROPEAN UNION



Mannarini et al 2021, https://doi.org/10.3390/jmse9020115

cmcc

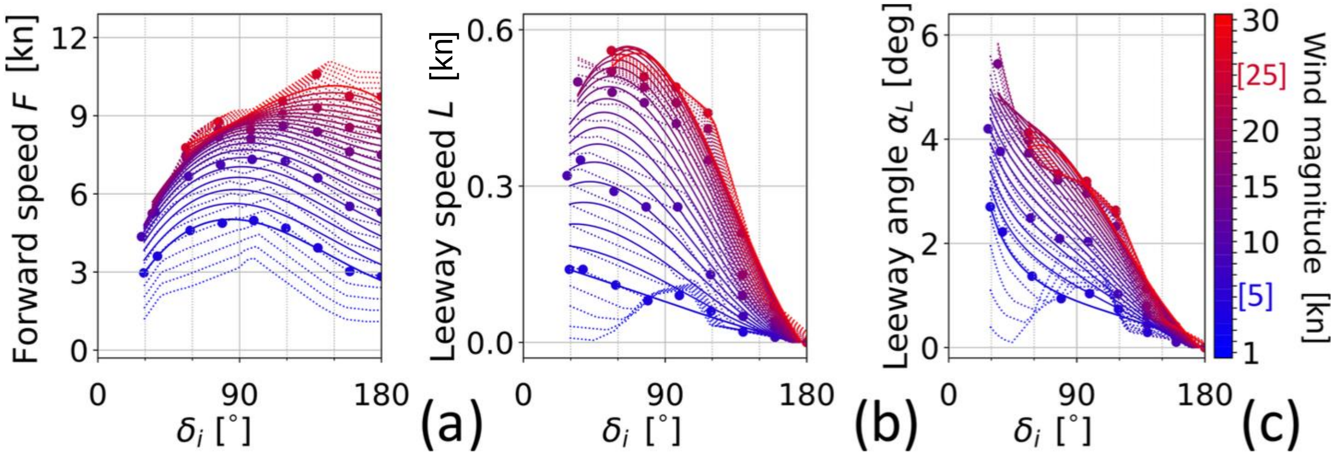
Vessel performance curve: sailboat



WinDesign velocity prediction program run at University of Genoa:

- *considered wave added resistance by means of "Delft method"*
- *response amplitude operators derived from CFD*
- *choice of sail:*
  - *upwind: main sail and jib sail*
  - *otherwise: main sail and spinnaker*

Interreg Italy - Croatia FRAMESPORT — EUROPEAN UNION



no-go angle varies from 27 to 53° as wind speed increases from 5 to 25 kn

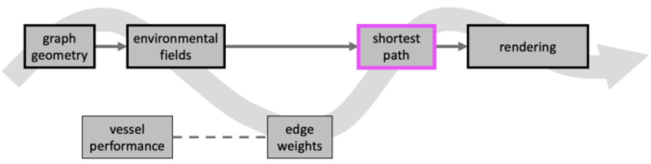
*forward speed **F** increases with wind intensity*

*peak **F** attained for broad reach ($\delta_i \approx 135°$)*

Leeway is max for points of sail between no-go zone and beam reach ($\delta i = 90°$)

As the point of sail from the no-go zone to running conditions, leeway angle reduces from 6 to 0°

Dynamic least-CO_2 algorithm



Dijkstra's algorithm generalized for dynamic edge weights

under FIFO hypothesis, same complexity of a static algorithm

built on the `single_source_Dijkstra` *function of the* `networkX` *library*

use of data structures (heaps) to achieve ideal performance

key advancement for least-CO2 paths is accessing edge weight at a specific time step

Algorithm 2 GET_TIME_INDEX.

Require: (*paths*, *d*, *wT*, *Ntau*, *Dtau*), a dictionary of paths, node costs, type of edge weight, maximum number of time steps, and time resolution, respectively
Ensure: *t_idx*, the time step at which the costs *d* are realised along the *paths*

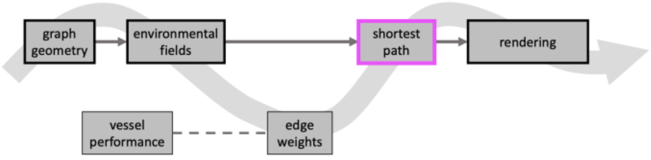
if wT = "*time*" **then**
  $t\_idx \leftarrow \min(Ntau, \lfloor d/Dtau \rfloor)$
else
  # compute cTime cumulative time
  $cTime \leftarrow 0$
  $t\_idx \leftarrow 0$
  for *edge* **in** *paths* **do**
    # evaluate edge delay at time step t_idx
    $cTime \leftarrow cTime + edge.cost.at\_time(t\_idx, \text{"time"})$
    $t\_idx \leftarrow \min(Ntau, \lfloor time/Dtau \rfloor)$
  end for
end if

Algorithm 1 _DIJKSTRA_TDEP.

Require: *(G, source, target, wT, Ntau, Dtau)*, a `NetworkX` graph, *source* and *target* nodes, type of edge weight, maximum number of time steps, and time resolution, respectively
Ensure: *(costs, paths)*, Two dictionaries keyed by node ID: path costs from the source (e.g. cumulated CO_2) and corresponding optimal paths

costs ← {}
seen ← { *source*:0 }
paths ← { *source:[source]* }
 # fringe is a min-priority queue of (cost, node) tuples
fringe ← *heap()*
fringe.push(0, source)
while $fringe \neq \varnothing$ **do**
  $(d, v) \leftarrow fringe.pop()$
  if $v \in costs$ **then**
    # Already visited node
    skip
  end if
  costs[v] ← *d*
  if v = target **and** $\forall n \in G.neigh(target), n \in seen$ **then**
    exit
  end if
  $t\_idx \leftarrow get\_time\_index(paths[v], d, wT, Ntau, Dtau)$
  # Iterate on v's forward-star
  for *(u, cost)* in *G.succ(v)* **do**
    # evaluate edge weight of wT type at time step t_idx
    $c \leftarrow cost.at\_time(t\_idx, wT)$
    $vu\_cost \leftarrow costs[v] + c$
    if $u \notin seen$ or $vu\_cost < seen[u]$ **then**
      $seen[u] \leftarrow vu\_cost$
      $fringe.push(vu\_cost, u)$
      $paths[u] \leftarrow paths[v] + [u]$
    end if
  end for
end while

Validation



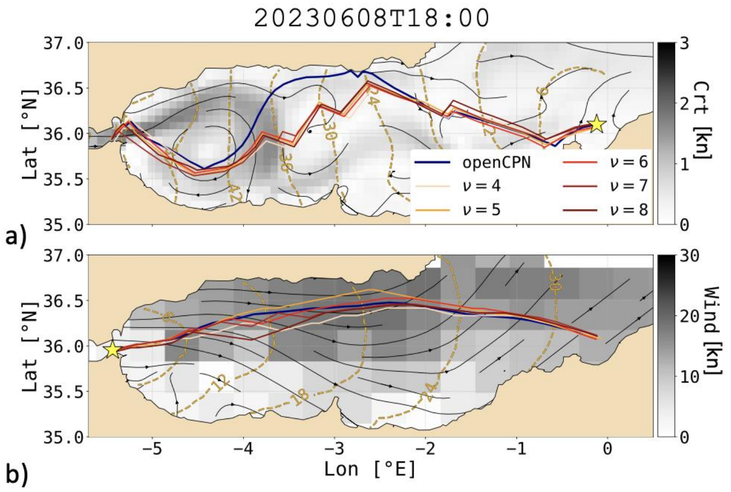
VISIR-2 routes and metrics were compared to:

- *semi-analytical results (cycloid, Techy)*
- *MIT model based on partial differential equations (LSE, *)*
- *openCPN (dynamic programming)*

Oracle	ν	$1/\Delta x$ $1/°$	$\Delta\tau$ min	L_0 nmi	T_0 h	$T^{(e)}$ T_0	T^* T_0	dT^* %
Cycloid	2	60	5				1.738	0.691
	5	60	5	56.38	2.672	1.726	1.726	0.012
	10	50	5				1.732	0.342
Techy	5	25	5				1.057	0.076
	5	100	5	140.11	6.640	1.056	1.046	−0.956
	10	50	5				1.050	−0.599

1 nmi = 1852 m.

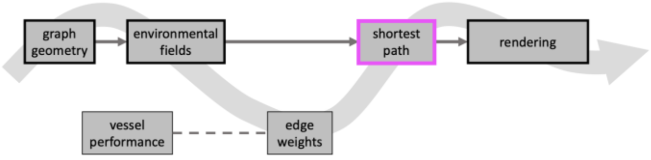
Model	ν	$1/\Delta x$ $1/°$	$\Delta\tau$ min	T^* h	dT^*_{120} %	dT^*_{240} %
VISIR-1.b	6	129	–	13.73	−1.58	−0.43
VISIR-2	6	129	30	13.62	−2.36	−1.23
VISIR-1.b	3	134	–	13.79	−1.12	0.03
VISIR-2	3	134	30	13.71	−1.73	−0.59
VISIR-1.b	2	142	–	13.90	−0.36	0.79
VISIR-2	2	142	30	13.85	−0.74	0.42
LSE	–	120	–	13.95	–	–
	–	240	–	13.79	–	–



			Wind				Current + wind			
			Westbound		Eastbound		Westbound		Eastbound	
Version	ν	$1/\Delta x$ [1/°]	T^* [h]	dT^* [%]	T^* [h]	dT^* [%]	T^* [h]	dT^* [%]	T^* [h]	dT^* [%]
VISIR-2	4	12	51.9	3.4	34.4	−0.4	54.0	−2.6	32.2	0.0
	5	15	52.0	3.5	34.5	−0.2	53.9	−2.7	31.7	−1.5
	6	18	51.2	1.9	33.6	−2.9	53.4	−3.7	30.9	−3.9
	7	21	50.7	1.0	32.8	−5.0	52.8	−4.8	30.9	−4.1
	8	23	51.0	1.6	32.8	−5.0	53.1	−4.2	30.8	−4.5
OpenCPN			50.2		34.6		55.4		32.2	

**) Mannarini et al 2019, doi.org/10.1109/TITS.2019.2935614*

Numerical performance: optimal paths



Three variants of the algorithm:

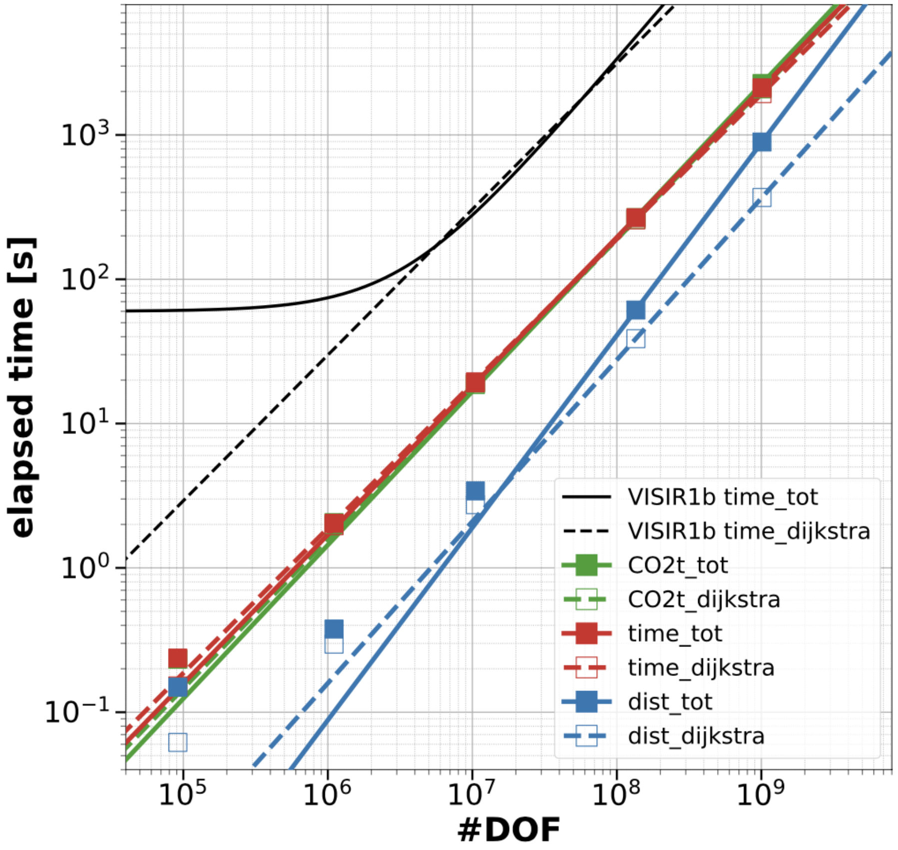
- *least-distance*
- *least-time*
- *least-CO2*

Assessment for:

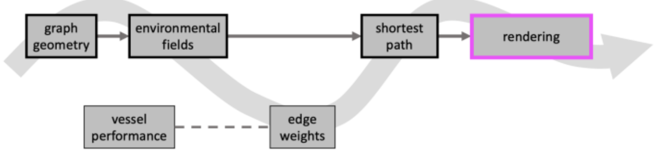
- *"Dijkstra": optimal sequence of graph nodes*
- *"total" : "Dijkstra" + marine and vessel dynamical information along the path*

Outcome:

- *linearity in the number of DOF*
- *10x faster than VISIR-1*
- *least-distance routine still to be improved*
- *RAM:* `420B` *per DOF (5x more than VISIR-1, to be improved e.g via single precision)*



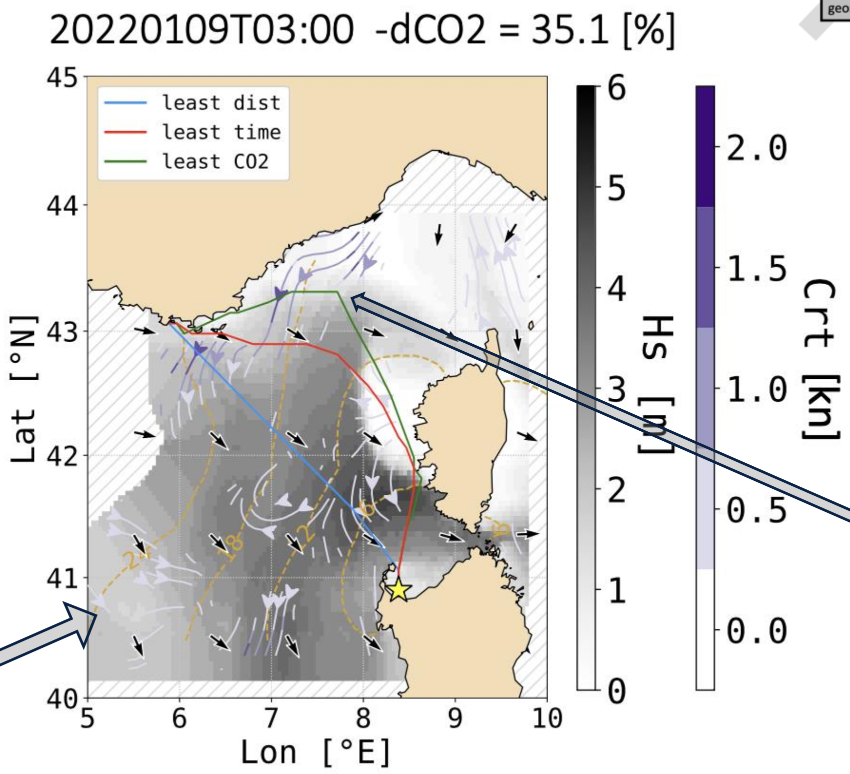
Visualisation



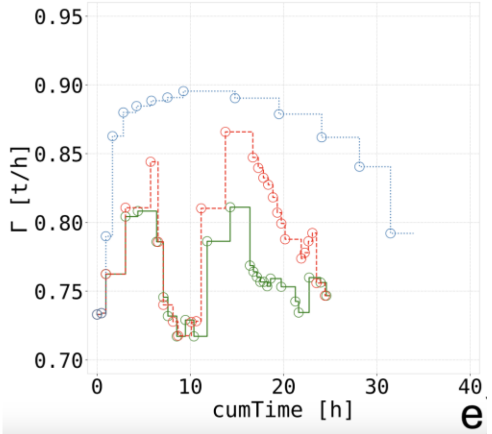
dynamic environmental fields rendered via

- *concentric shells originating at the departure location*
- *shape of shells defined by isochrones*

saving of 1 dimension (can be used for departure date or engine load)



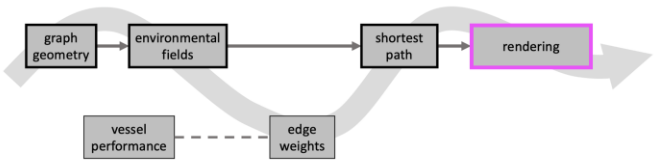
least-CO2 route takes a larger diversion than least-time to sail where emission rate is lower:

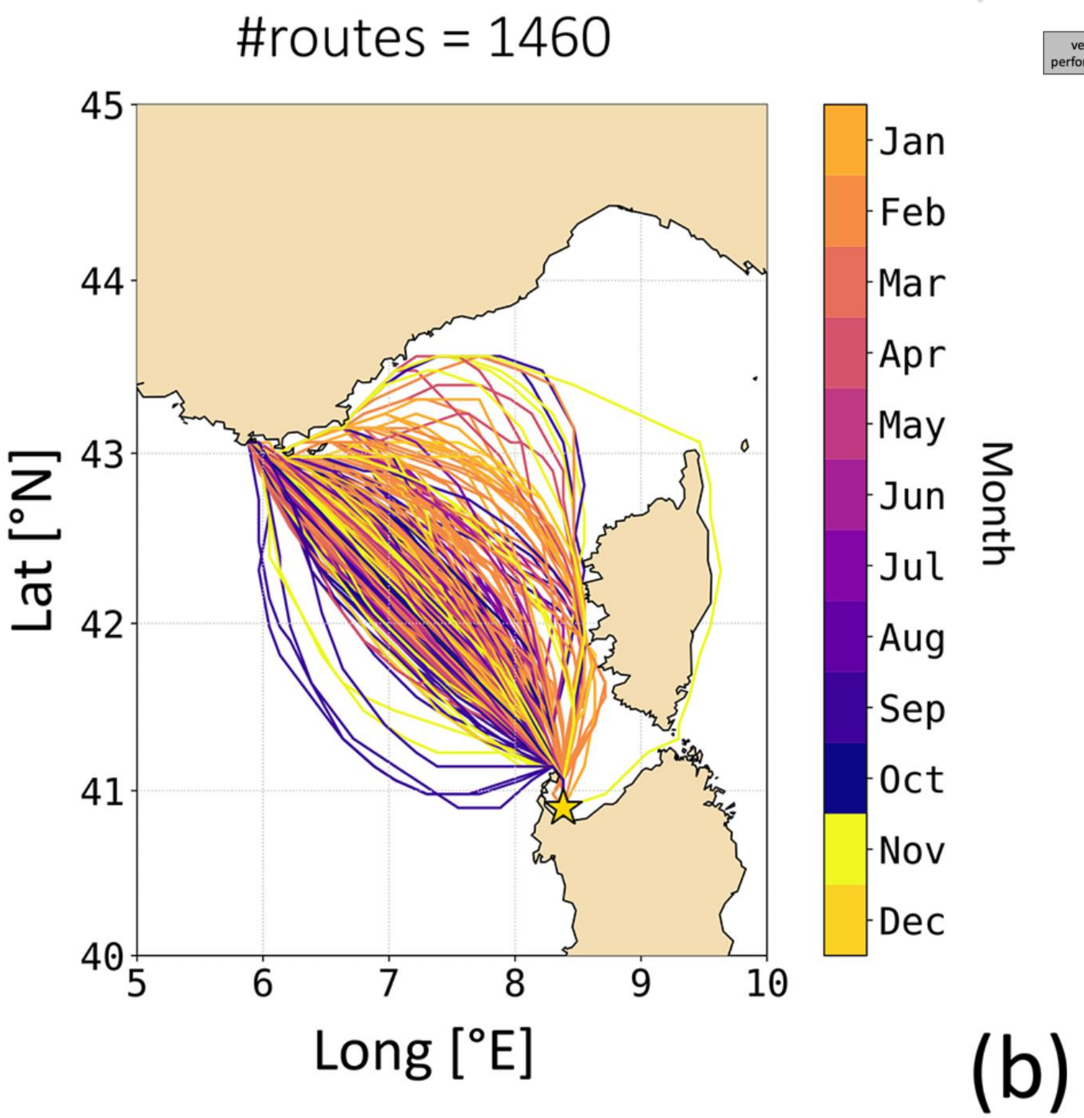


type	*meaning*	*bulging*
isometres	*equal distance*	*at obstructions (shoals, islands, landmass in general)*
isochrones	*equal duration*	*against gradients of 1/STW*
isopones	*equal emissions*	*against gradients of emissions*

cmcc

Visualisation





bundles of least-CO2 routes:

- *route colour by departure month*
- *seasonal dependence*
- *topological change is possible*

For the route east of Corsica, see Fig. S8-S9 of *https://doi.org/10.5194/gmd-17-4355-2024-supplement*

cmcc

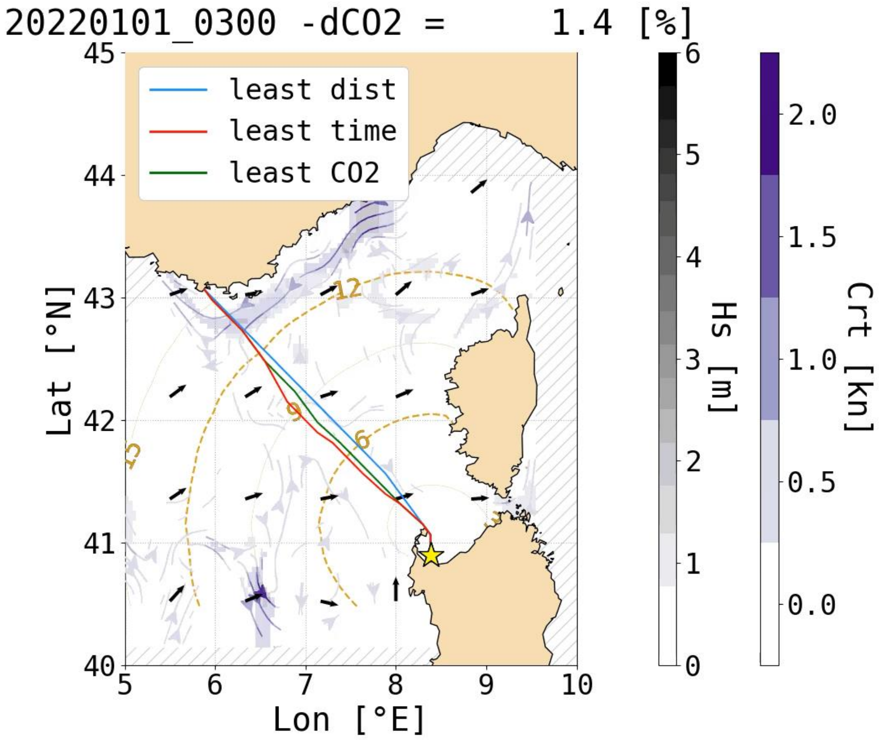
Outline

Case study: ferry

dynamic environmental fields rendered via

- *concentric shells originating at the departure location*
- *shape of shells defined by isochrones*

saving of 1 dimension (can be used for departure date or engine load)



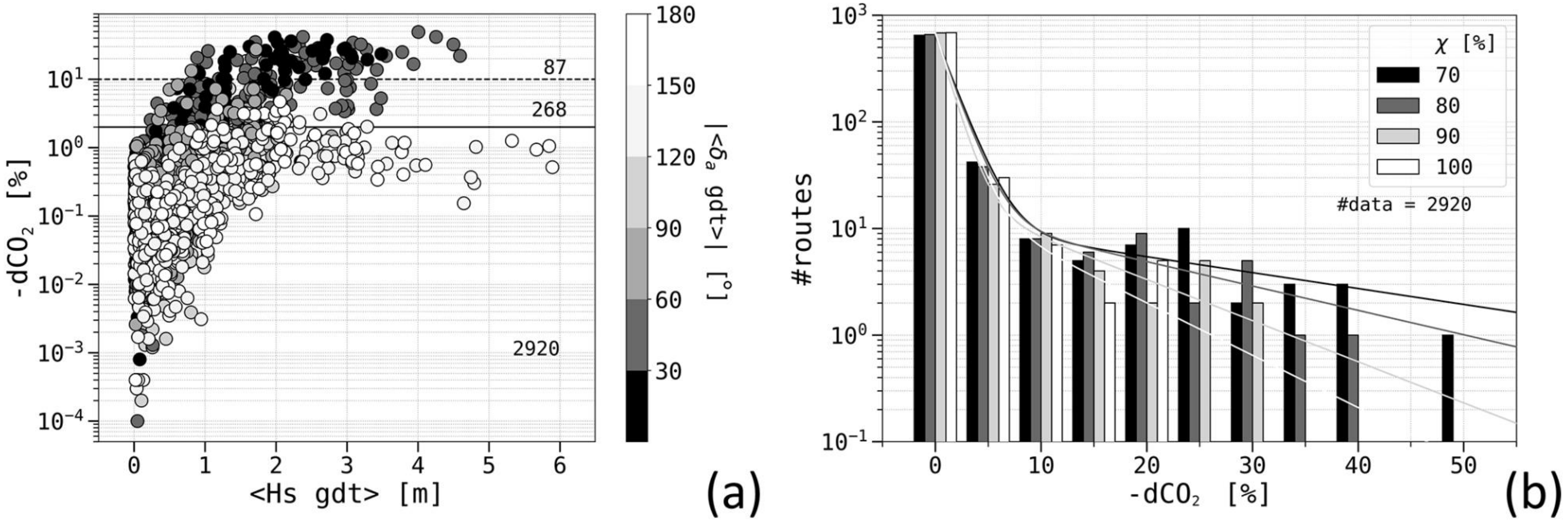
movies with 1-year of daily departures : https://doi.org/10.5446/s_1687

type	*meaning*	*bulging*
isometres	*equal distance*	*at obstructions (shoals, islands, landmass in general)*
isochrones	*equal duration*	*against gradients of 1/STW*
isopones	*equal emissions*	*against gradients of emissions*

cmcc

Case study: ferry

*Statistics of **CO2 savings** in 2022*



	upwind					*downwind*				
	ITPTO–FRTLN χ [%]					FRTLN–ITPTO χ [%]				
	70	80	90	100	Avg	70	80	90	100	Avg
wa	2.9	2.2	1.4	1	1.9	0.9	0.6	0.4	0.3	0.6
wa-cu	3.4	2.5	1.7	1.2	2.2	1.4	1.2	0.7	0.6	1

- *largest savings are upwind*
- *currents increase savings, especially downwind*

- *wave height alone is not the key to minimal savings*
- *key is angle of attack of waves*
- *> 2% for beam or head seas*
- *>10% for about 10 days in a year*

- *bi-exponential distribution*
- *larger decay length inversely proportional to engine load χ*
- *tail can extend to values ranging between 25 and 50%*

Case study: sailboat

movies with 1-year of daily departures : [*https://doi.org/10.5446/s_1688*](https://doi.org/10.5446/s_1688)

Geography

- *Meltemi wind*
- *Asia minor current*
- *archipelagic domain*

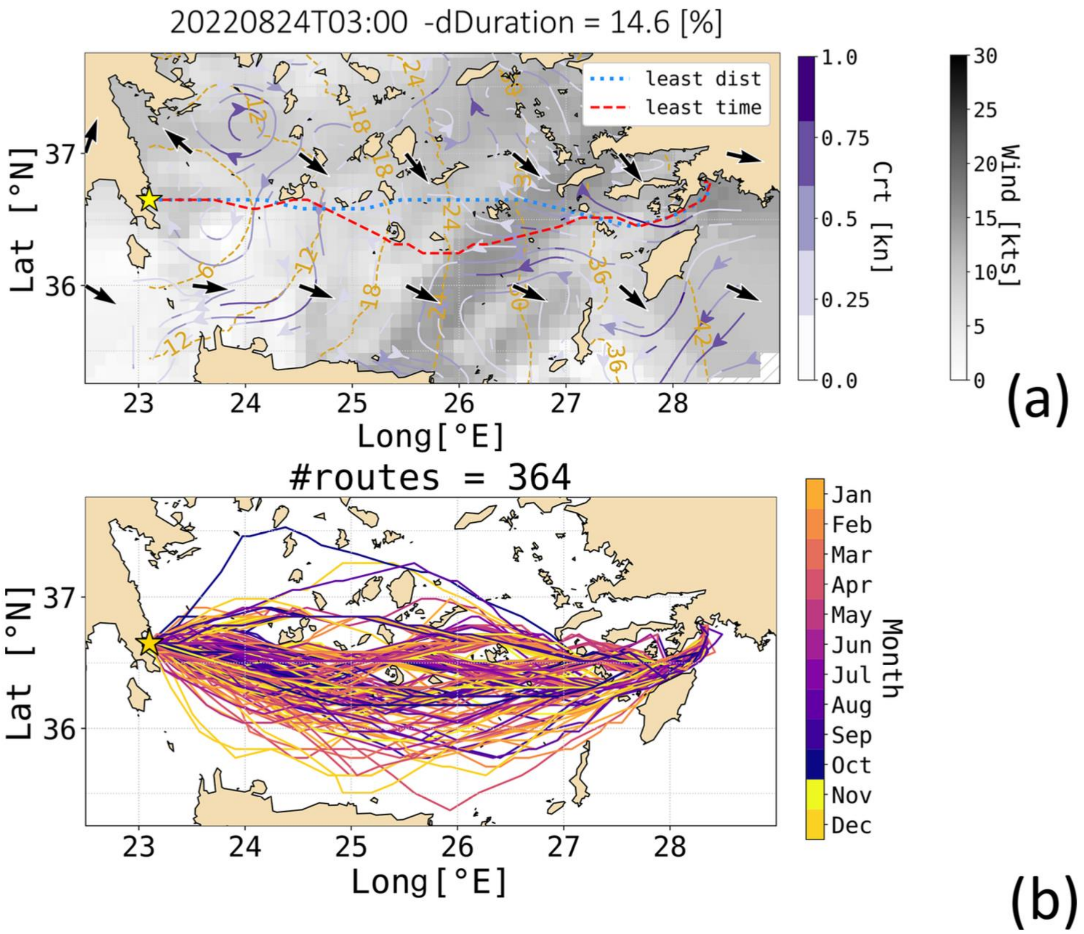
Name	Symbol	Value	Units
Length of hull	L_{hull}	10.7	m
Draft	T	2.2	m
Displacement volume	∇	5.8	m^3
Rudder wetted surface	–	1.4	m^2
Keel wetted surface	–	3.3	m^2
Main sail area	–	38	m^2
Jib sail area	–	4	m^2
Spinnaker area	–	95	m^2

Numerical experiments

- *graph with* $(v, 1/\Delta x) = (5, 15/^o)$
- *daily departures, two orientations, with/without currents or leeway (2,920 runs)*
- *7 min/run*

Outcome

- *large diversions to avoid upwind sailing*
- *no clear seasonal trend for diversions*



Case study: sailboat

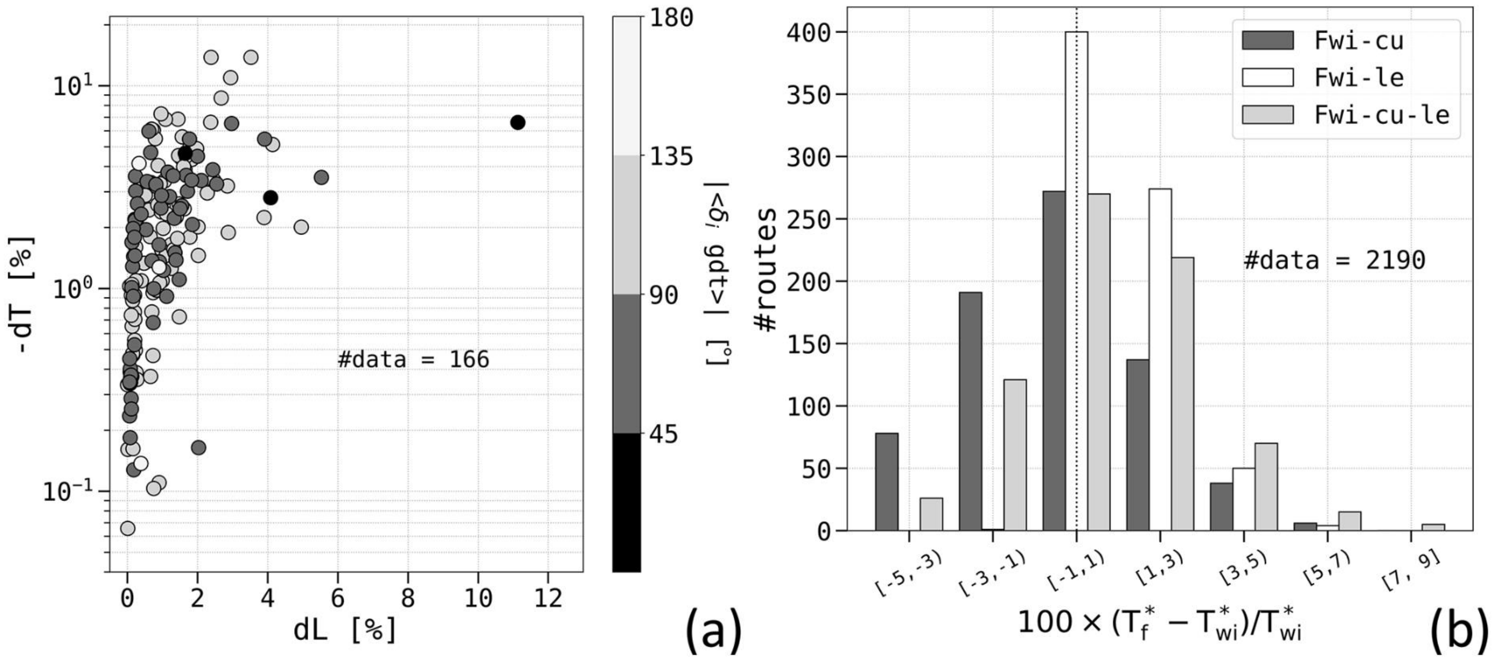
*Statistics of **time savings** in 2022*

downwind — *upwind*

against current — *with current*

	GRMON–TRMRM			TRMRM–GRMON		
	$-dT^*$	$N_f^{(g)}$	$N_f^{(o)}$	$-dT^*$	$N_f^{(g)}$	$N_f^{(o)}$
wi	2.5	256	1	2.3	308	1
wi-le	2.3	275	1	2.4	328	3
wi-cu	3.1	254	1	3.1	320	1
wi-cu-le	3.2	267	0	3.5	326	6

- *consideration of currents increases duration percentage savings*
- *leeway results in more failed routes (due to vessel manoeuvrability)*



- *time saving increases with spatial diversion*
- *max diversions when skipping upwind along geodetic route*

- *currents results in a change in duration (up to about 5%, skewed to reduction)*
- *leeway: consistently extends the duration (leeway cross-course component reduces SOG)*

Operational services based on VISIR-2

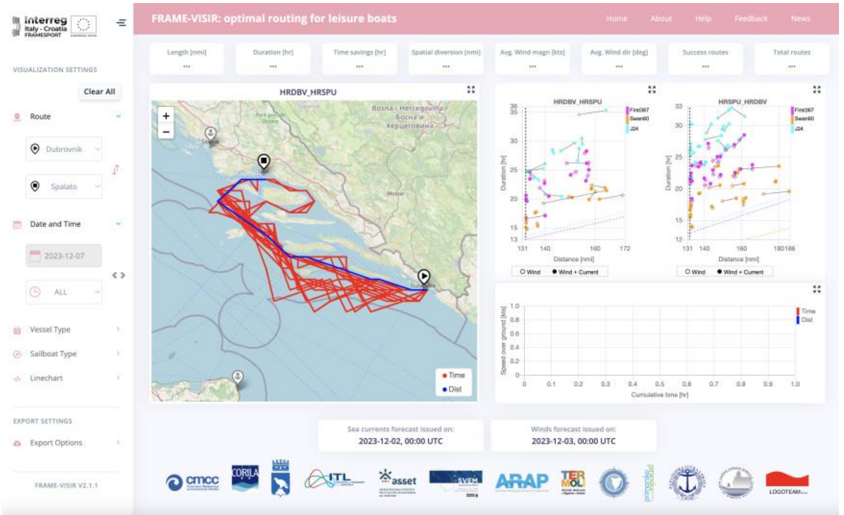
GUTTA-VISIR

https://www.gutta-visir.eu

- *ferry routes*
- *CO2 and CII savings*
- *cross-border routes*
- *7,020 routes/day*
- *operational since Oct.2021*

FRAME-VISIR

https://www.frame-visir.eu



- *sailboat and flybridge routes*
- *time and CO2 savings*
- *cross-border and cabotage routes*
- *5,280 routes/day*
- *operational since May 2023*

cmcc

Outline

1. *Introduction*
2. *Model workflow*
3. *Case studies*
4. *Discussion*

Results

- ✓ *VISIR-2: a modular, validated, documented, and portable model for ship weather routing*
- ✓ *for vessels with an angle-dependent performance curve, an improved level of accuracy in the velocity composition with sea currents*
- ✓ *a variant of the Dijkstra's algorithm developed (minimising not just the CO_2 emissions but any figure of merit depending on dynamic edge weights)*
- ✓ *quasi-linear computational performance up to 1 billion DOF*
- ✓ *10x faster than VISIR-1*
- ✓ *Bi-exponential distribution of CO_2 savings found for a ferry, with savings at times exceeding 10%*
- ✓ *sailboat routes: duration savings of about 3% , neglecting leeway wrongly underestimates route durations*

Novelty

- *wave, winds, and currents addressed through an unified scheme*
- *role of currents assessed*
- *versatile least-CO_2 algorithm (can become a least-X algorithm)*
- *shape of statistical distribution of CO_2 savings assessed*
- *consideration of both currents and leeway in the optimization of sailboat routes*
- *visualisation making use of isochrone-bounded sectors*

VISIR-2 resources

open access - open review journal paper

https://doi.org/10.5194/gmd-17-4355-2024

Geosci. Model Dev., 17, 4355–4382, 2024
https://doi.org/10.5194/gmd-17-4355-2024

Geoscientific
Model Development
Open Access

Model description paper

VISIR-2: ship weather routing in Python

Gianandrea Mannarini[1], Mario Leonardo Salinas[1], Lorenzo Carelli[1], Nicola Petacco[2], and Josip Orović[3]

[1]CMCC Foundation – Euro-Mediterranean Center on Climate Change, Lecce, Italy
[2]DITEN, Università degli Studi di Genova, via Montallegro 1, 16145 Genoa, Italy
[3]Maritime Department, University of Zadar, Ul. Mihovila Pavlinovića, 23000 Zadar, Croatia

open language - open source model code

https://zenodo.org/communities/visir2

April 11, 2024 (v3) | Software | Open

[VISIR-2 ship weather routing model] source code (Python)

Salinas, Mario Leonardo; Carelli, Lorenzo; Mannarini, Gianandrea

Source code of the VISIR-2 ship weather routing model. The Python-refactored VISIR-2 model considers currents, waves, and wind to optimise routes. The model was validated, and its computational performance is quasi-linear. For a ferry sailing in the Mediterranean Sea, VISIR-2 yields the largest percentage emission savings for upwind navigation...

Part of VISIR-2 ship weather routing model (Python)

Uploaded on April 11, 2024

2 more versions exist for this record

cmcc

Centro Euro-Mediterraneo sui Cambiamenti Climatici

www.cmcc.it