Nat. Hazards Earth Syst. Sci., 16, 1791–1806, 2016
www.nat-hazards-earth-syst-sci.net/16/1791/2016/
doi:10.5194/nhess-16-1791-2016

Natural Hazards and Earth System Sciences
Open Access
EGU

VISIR: technological infrastructure of an operational service for safe and efficient navigation in the Mediterranean Sea

Gianandrea Mannarini[1], Giuseppe Turrisi[1], Alessandro D'Anca[2], Mario Scalas[3], Nadia Pinardi[1,4], Giovanni Coppini[1], Francesco Palermo[1], Ivano Carluccio[1], Matteo Scuro[1], Sergio Cretì[1], Rita Lecci[1], Paola Nassisi[2], and Luca Tedesco[3]

[1]CMCC, Centro Euro-Mediterraneo sui Cambiamenti Climatici – Ocean Predictions and Applications, via Augusto Imperatore 16, 73100 Lecce, Italy
[2]CMCC, Centro Euro-Mediterraneo sui Cambiamenti Climatici – Advanced Scientific Computing, Strada Provinciale per Arnesano (complesso Ekotekne), 73100 Lecce, Italy
[3]Links Management and Technology S.p.A. – via R. Scotellaro 55, 73100 Lecce, Italy
[4]Università degli Studi di Bologna, viale Berti-Pichat, 40126 Bologna, Italy

Correspondence to: G. Mannarini (gianandrea.mannarini@cmcc.it)

Received: 5 February 2016 – Published in Nat. Hazards Earth Syst. Sci. Discuss.: 10 February 2016
Revised: 14 July 2016 – Accepted: 14 July 2016 – Published: 3 August 2016

Abstract. VISIR (discoVerIng Safe and effIcient Routes) is an operational decision support system (DSS) for optimal ship routing designed and implemented in the frame of the TESSA (TEchnology for Situational Sea Awareness) project. The system is aimed to increase safety and efficiency of navigation through the use of forecast environmental fields and route optimization. VISIR can be accessed through a web interface (www.visir-nav.com) and mobile applications for both iOS and Android devices. This paper focuses on the technological infrastructure developed for operating VISIR as a DSS. Its main components are described, the performance of the operational system is assessed through experimental measurements, and a few case studies are presented.

1 Introduction

Targeted services based on operational outputs from geophysical models represent a new paradigm of knowledge. In fact, for enabling operational use even by decision makers with a limited knowledge of the underlying natural processes, these services require technological tools of increasing complexity and robustness (Hey et al., 2009). Furthermore, the reproducibility and the degree of objectiveness of these services critically depend on the traceability of the computational process and the quality of the computer codes behind it (Weintrit et al., 2013).

In the framework of maritime safety and efficient transportation, situational sea awareness through the operational distribution of oceanographic and meteorological information is a key enabler of technological applications. In fact, the use of marine weather forecasts for route recommendations has been since long recognized (Bowditch, 2002).

However, due to the limited spatial and temporal resolution of the oceanographic forecast products, so far applications have dealt mainly with large ocean-going motor vessels or racing and leisure sailboats, mainly in a regime of open-sea navigation. In recent years, the operational availability of coastal observatories and high-resolution ocean forecast products have paved the way for applications to be used even in enclosed seas and coastal waters (Proctor and Howarth, 2008).

A list of both institutional and commercial ship routing systems is provided by Lu et al. (2015), and some major commercial softwares are compared in Walther et al. (2014). A few other systems are here reviewed with an emphasis on their operational functioning.

Montes (2005) reported on a model for automatizing the Optimum Track Ship Routing (OTSR) service provided by the US Navy to its own ships. It is a least-time routing system based on wave forecasts, whereby the safety of navigation is accounted for through speed penalty functions related to the sea state. Interestingly, model outputs are validated versus historical records of route diversions by the US war ships in

the western Pacific Ocean. However, the related operational system is not publicly accessible.

A new concept of Sea Traffic Management (STM) has been developed during the MONALISA series[1] and ACCSEAS[2] European projects (Fiorini and Lin, 2015) and is going to be implemented in the coming years, starting with the STM Validation Project[3]. In the current definition of STM, the marine voyage is the central object of analysis and development (Siwe et al., 2015). A common format (RTZ[4]) and architecture for a seamless exchange of route information and voyage plans was designed and standardized during the MONALISA 2.0 project. In the STM framework, voyage management services will provide support to individual ships in both the planning process and during the voyage, also making use of route optimization services.

The Finnish transport agency developed ENSI[5], a system for providing ships navigating in the Gulf of Finland with a shore-based support. The route is first planned onboard using an ECDIS; then it is broadcast to a vessel traffic centre, where it is validated against topological and marine weather information, and finally it is broadcast back to the ship. Warning items are issued if the shore-based centre detects that the onboard planned route is either going to cross a traffic separation scheme or sail over shallow water. Since spring 2016, routes have been exchanged by ENSI following the standard RTZ format.

Promising work has been performed at NTNU for short-sea routing of vessels supplying offshore installations (Kjølleberg, 2015). The model takes into account wave forecasts for computing the optimal sequence for visiting oil platforms, with the objective of optimizing some cost function. The model does not seem to be part of an operational system yet, though it is worth noticing that its code has been made open source.

To the best of our knowledge, an open-access, operational ship routing system using state-of-the-art meteo-oceanographic forecasts for route optimization had been missing before we, in the frame of an Italian national project aimed at technological transfer (TESSA: TEchnology for Situational Sea Awareness), developed VISIR ([vi'zi:r], discoVerIng Safe and effIcient Routes)[6]. VISIR is meant to provide an open-access (but not necessarily free), user-oriented, cross-channel system for on-demand computation of optimal routes for various kinds of both motor- and sailboats navigating the Mediterranean Sea. In VISIR-I, the first version of the system, optimization regards the total time of navigation, taking into account safety of navigation.

In this paper we mainly report on the technological infrastructure developed for operating VISIR as a decision support system (DSS). However, we deem it useful to first provide a compact introduction to the VISIR ship routing model in Sect. 2. The description of the operational system is split into a more infrastructural perspective given in Sect. 3 and a functional one in Sect. 4. In Sect. 5 two case studies, for motorboat and sailboat routing respectively, are presented. The conclusions given in Sect. 6 summarize the authors' experience gained while realizing VISIR and briefly discuss the more general perspectives for this kind of technological infrastructure.

2 The ship routing model

The model behind the VISIR operational system has been developed from scratch in the frame of the TESSA project. Its numerical structure is described in highest detail in Mannarini et al. (2016b). The model is presently coded in MATLAB.

VISIR-I, the first version of the model, employs meteo-oceanographic forecast products to optimize nautical routes. The optimization objective is to keep the total sailing time at an absolute minimum while treating safety of navigation as a constraint. Safety includes, for both motor- and sailboats, consideration of minimum under-keel clearance (UKC) and, just for motorboats, dynamical stability checks too. At each VISIR-I run, both a geodetic and an optimal route are computed. The geodetic route is a least-time route not taking into account the dynamic environmental conditions but still complying with the static safety constraints related to the bathymetry and the shoreline. The optimal route takes into account the dynamic environmental conditions both to compute vessel kinematics and, just for motorboats, to avoid the dynamical hazards of navigation. VISIR-I as a model consists, as reported in the following subsections, of three main components: the environmental forecasts, the optimization algorithm, and the vessel model. The MATLAB source code of VISIR-I is made free and open access under a GPL licence[7] (Mannarini et al., 2016a).

2.1 Environmental forecasts

In VISIR-I both sea-state (waves) and wind forecasts are employed: the sea-state information is used for motorboat routing and the wind information for sailboat routing.

Small- and middle-sized motorboats are considered by VISIR-I. Mannarini et al. (2016b) discussed the most relevant environmental couplings for this class of vessels, concluding that waves are likely to be the phenomenon with the most impact. Thus, VISIR-I employs wave forecast fields from an operational implementation of the Wave Watch III (WW3) model in the Mediterranean Sea (Tolman, 2009). The

[1] http://monalisaproject.eu/

[2] http://www.accseas.eu/

[3] http://stmmasterplan.com/

[4] IEC 61174, edition 4: https://webstore.iec.ch/publication/23128

[5] https://www.ensi.fi/portal/

[6] http://www.visir-nav.com/

[7] http://www.visir-model.net/

employed fields are significant wave height, mean[8] wave period, and wave direction. They are provided with hourly resolution on a 1/16 degree (i.e. 3.75 M in the meridional direction) mesh. The forecasts originating from the 12:00 UTC analysis are employed.

For sailboats, 10 m height wind forecasts from IFS model[9] operated by the European Centre for Medium-Range Weather Forecasts (ECMWF) are employed. The model outputs are available with 3-hourly resolution for the first 3 days after the analysis, the horizontal resolution is 1/8 degree (7.5 M in the meridional direction, M = nautical mile), and the forecasts refer to the 12:00 UTC analysis.

2.2 Optimization algorithm

VISIR-I's optimization is based on a graph-search algorithm. Dijkstra's algorithm (Dijkstra, 1959) is chosen for its ease of implementation and the fact that, not depending on any heuristics, it is guaranteed to find an optimal solution. Furthermore, since the environmental forecasts are represented by time-dependent fields, the algorithm is modified along the guidelines by Orda and Rom (1990) for ingesting such dynamic information. Also, VISIR's algorithm allows for voluntary ship speed reduction. This option enables the ship to not change course for ensuring the vessel stability constraints, resulting in additional savings of navigational time. All the mathematical details, the algorithm's validation, and its pseudocode are provided in Mannarini et al. (2016b). The target grid for the optimization is a 1/60° (1 M in the meridional direction) regular mesh, and the connectivity of the graph is such that angles of about 27° can be resolved. Finally, thanks to its masking procedure, VISIR is capable of avoiding the land mass even in topologically complex areas, such as peninsulas, islands, and archipelagic seas.

2.3 Vessel model

Two quite different approaches are adopted for modelling the dynamical response of either motor- or sailboats to the environmental conditions, as shortly summarized in the following.

For motorboats, just displacement vessels (fishing vessels, service boats, displacement hull yachts, and small ferry boats) are considered in VISIR-I. Sustained vessel speed is obtained from a balance between the thrust provided by the propeller and the total resistance applied to the moving hull in any given sea state. In order to reduce the number of parameters to be set by the end-user, the motorboat vessel model is kept simple, neglecting several mechanical effects affecting vessel dynamics; see Mannarini et al. (2016b) for more details. The six parameters to be provided by the end-user are listed in Table 1. Furthermore, vessel stability in a seaway is considered (IMO, 2007). In particular, the dynamical conditions for the activation of three stability loss mechanisms are checked for: parametric roll, pure loss of stability, and surfriding/broaching (Belenky et al., 2011). Graph edges leading, for a specific time step, to stability loss are removed from the graph prior to running the optimization algorithm. This way, it is ensured that the optimal route does not result in an exposure to dynamical hazards.

Table 1. Parameters of the motorboat model and their values for the case study of Fig. 7.

Symbol	Name	Units	Typical value
P	Actually delivered engine power	hp	650
c	Top speed	kt	10.7
L	Length at the waterline	m	22
B	Beam (width at waterline)	m	6
T	Draught	m	2
T_R	Natural roll period	s	5.4

Sailboats are described in terms of their "polar plots". These are response functions expressing sailboat speed in terms of wind speed and direction. They stem from either measured sailboat performance or so-called velocity prediction programmes (de Jong et al., 2008). In VISIR-I, polar plots for a given and fixed sail are considered. Each polar plot contains a no-go zone, accounting for the fact that direct navigation into the wind is not possible. More on this subject can be found in Mannarini et al. (2015).

3 The operational infrastructure

In order to make the ship routing system of Sect. 2 operational, a hardware and software infrastructure has been built in the frame of the TESSA project. In addition to VISIR, TESSA also supported the development of several other DSSs using the outputs of meteo-oceanographic models: for instance, an oil spill management DSS[10] (Liubartseva et al., 2016) and a DSS for supporting marine search-and-rescue operations[11] (Coppini et al., 2016a; Jansen et al., 2016) were also developed. All the input meteo-oceanographic model outputs can be accessed from the Sea-Conditions[12] (Coppini et al., 2016b) portal. VISIR as an operational system inherits the infrastructural approach by TESSA, which is a web-based architecture, exposing multichannel functionality and decoupling the service front and back end.

Being multichannel implies that the service is made available across different web and mobile platforms (desktop computers, tablets, smartphones). Furthermore, the services

[8] Peak period was used instead of mean wave period until 13 June 2016.

[9] http://www.ecmwf.int/en/forecasts/documentation-and-support/evolution-ifs/scorecards/scorecard-ifs-cycle-40r1

[10] http://www.witoil.com/

[11] http://www.ocean-sar.com/

[12] http://www.sea-conditions.com/

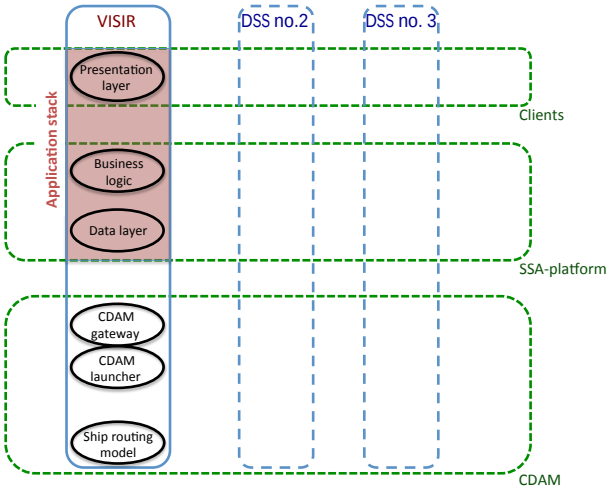

Figure 1. "TESSA matrix": the three horizontal tiers and the vertical applications or DSSs. The red shaded area is the VISIR application stack.

are developed keeping in mind the "3C" paradigm. That is, the user is granted a consistent, continuous, and complementary experience, through an ecosystem of platforms where the same service is made available (Levin, 2014). This is realized through the decoupling between service front and back end, enabling a thin client to drive complex data analysis on a supercomputing facility.

From a structural viewpoint, the TESSA architecture is a "matrix" of tiers and vertical applications. The tiers are the clients, the "Situational Sea-Awareness (SSA) platform", and the "Complex Data Analysis Module" (CDAM). The vertical applications correspond instead to the various DSSs (see Fig. 1).

The client tier allows the end-user to send commands to and receive results from the SSA platform. The SSA platform bridges a bidirectional communication between the clients and the CDAM and provides maps of environmental fields to the client tier. The CDAM manages the incoming computation requests, runs the model (e.g. the one for ship routing), and returns the results. Furthermore, an application stack distributed between the client tier and the SSA platform provides the end-user with the tools required for interacting with the system.

Keeping in mind the block diagram of Fig. 2, the functioning of application stack, SSA platform, and CDAM are further detailed in the next three subsections.

3.1 The application stack

The VISIR application stack is a system component needed to receive the end-user's requests and deliver the ship routes to his/her device (see Sect. 7). As seen from Fig. 1, the application stack consists of components based on both the client tier and the SSA platform. It is structured as a multilayer architecture, which maps the system infrastructure to the physical layers on which the application is deployed and executed. In particular, the following three layers characterize the application stack: the presentation layer, the business logic layer, and the data layer.

The presentation layer is in charge of the visualization through the client rendering engine. This layer corresponds to the graphical user interface generally employed by desktop applications. The VISIR presentation layer is declined into a web application and two mobile applications. The web application (www.visir-nav.com) is a single-page application, providing universal, full-featured, cross-platform access to VISIR. It is coded in JavaScript and Java. The mobile applications have been implemented natively for each platform (i.e. in Objective C for iOS and in Java for Android). The rationale of creating native applications is to optimize visual performance, enhance platform-specific user experience, and, potentially, exploit best the hardware resources (definitively GPS, and possibly other device sensors such as the gyroscope and the accelerometer). For instance, a platform-specific type of map service is adopted: MapKit Framework[13] for Apple devices and Google Maps[14] for the web application and Android devices. Examples of the presentation layer are provided in Sect. 5. The mobile applications have been made available on the App Store and Google Play.

The business logic layer is the core part of the application stack and implements its logic. Its task is to receive, process and meet the incoming requests from the client. Several RESTful (Fielding, 2000)[15] services have been developed in order to manage these requests and to forward the required information to the other components of the infrastructure. The business logic layer is also responsible for exchanging information with the data layer.

The data layer is associated with the database engine and is responsible for the data persistence (within and across sessions[16]) and their querying. It receives and fulfils the database read/write requests coming from the business logic layer. In particular, by means of this layer, all input parameters related to the ship route computation and all the results produced by the VISIR model are stored in the database of the SSA platform.

While the presentation layer is platform-specific, the business logic layer and the data layer serve both the channels of the web application and the mobile applications.

[13] https://developer.apple.com/library/ios/documentation/MapKit/Reference/MapKit_Framework_Reference

[14] https://developers.google.com/maps/documentation/android-api/reference

[15] http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[16] A session is here defined as the user activity comprised between authentication and quitting from the DSS.

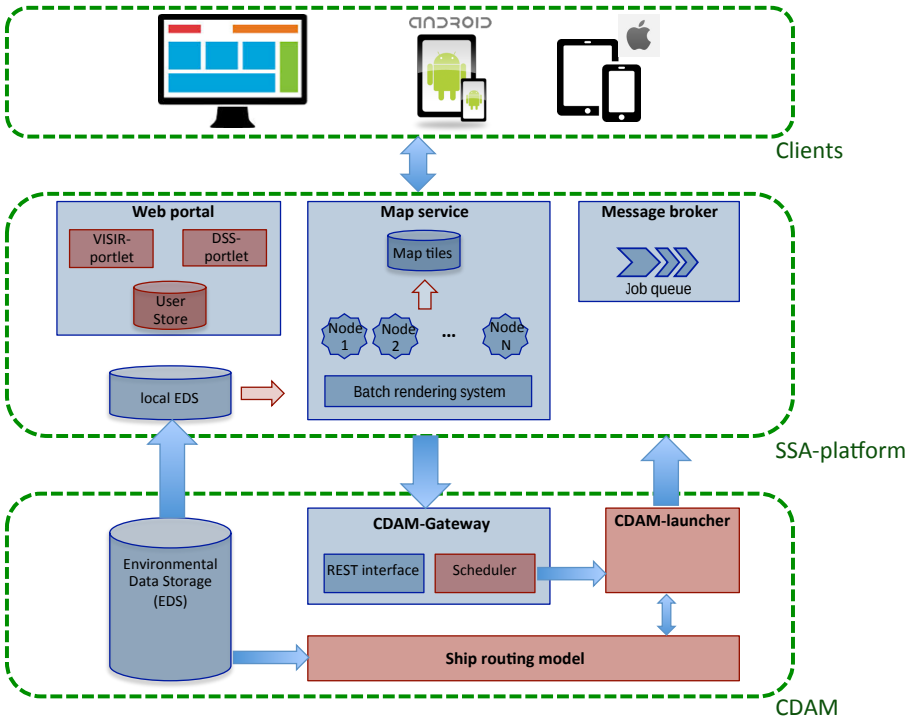

Figure 2. Functional structure of the TESSA system. Red boxes refer to VISIR-specific components hosted on the TESSA infrastructure (blue). Cylinders represent databases; boxes are services; icosagons are computational nodes.

3.2 The SSA platform

The SSA platform provides the infrastructure for processing the environmental forecast data and making the services available to public users across different channels, spanning from smartphones and web clients to (potentially) third-party geographic information systems (GISs) (Scalas et al., 2016). The SSA platform architecture can be divided into several components: a web portal, a map service, and a message broker (MB) (cf. Fig. 2). Communication between each of the clients and server-side software parts occurs according to open, standard protocols such as HTTP[17] and JSON[18].

The web portal is a customized web container hosting the applications and their web APIs and providing basic shared services like authentication and user management. Both the VISIR and the DSS portlet are hosted here. The VISIR portlet provides a universal user interface for any[19] browser. The DSS portlet provides the client tier with authentication and authorization policies and storage of the most recent computation results for later retrieval. All user credentials and permissions are tracked within the web portal by a user store: this allows fine-grained permissions and access settings to the web portal functionalities, like using a specific DSS or unlocking advanced features.

The map service is a cross-application component (used also by the other TESSA DSSs) designed to provide both static and dynamic maps and related functions (like data querying) to external services and applications. The map service is made up of a web service providing the maps, a batch rendering system updating the forecast maps on a daily basis, and a computing cluster performing the actual rendering work. The map service allows distribution of very large maps by delivering tiles of the environmental field at different scale as a set of 256-pixel-wide images ("tiles"), greatly improving efficiency in bandwidth and client resource usage. The indexing of these images has been defined in the past by different vendors, like Google and Microsoft[20], and then standardized by the Open Geospatial Consortium in the OpenGIS Web Map Tile Service (WMTS) 1.0.0 specifications[21]. Furthermore, the dynamic map service also provides an experimental WMTS service for improved interoperability with external GIS software. A RESTful API is provided to the clients for querying the available maps and accessing the data browsing functions (Sect. 2.1). The batch rendering system periodically fetches environmental data from the Environmental Data Storage (EDS) of the CDAM into a (SSA platform) local EDS and triggers their initial ingestion within the system (Fig. 2). The process also includes basic integrity checks and partial rendering of maps. That is, in order to

[17] http://www.w3.org/Protocols/
[18] http://www.JSON.org/
[19] Optimized for Firefox, Chrome, and Safari.
[20] http://www.maptiler.org/google-maps-coordinates-tile
[21] http://www.opengeospatial.org/standards/wmts

save resources, the batch rendering system pre-renders just a limited set of map tiles, allowing a rapid response for most frequently used maps. For the remaining tiles, an on-the-fly rendering is triggered, queueing the task to the computing cluster.

The MB is an intermediate component between the clients and the CDAM dispatching (DSS-specific) job requests to the heavy-duty computing back end, on behalf of the clients (either web or mobile). A call-back mechanism is provided as a hook to the CDAM in order to notify job completion, either successful or not, including the data payload. DSS components use this system in order to notify users about the results of their requests and perform additional actions (e.g. store the results into a historical archive for later retrieval). This software component acts as a store-and-forward queue, as it receives and stores requests, forwards them to the computing engine, and awaits a response. Clients may then retrieve the results by polling for their availability, checking the payload, and extracting the information they require for their work. Additionally, in order to avoid an excessive load, a data retention policy can be enforced (e.g. by setting a limited time before expiration of pending requests or removing old ones).

3.3 The Complex Data Analysis Module

The Complex Data Analysis Module (CDAM) enables advanced data processing on the datasets produced by the meteorological and oceanographic models used within the TESSA project. Specifically, it represents a back-end connector between the client-oriented services, deployed on the SSA platform, and the model execution related to a specific DSS (D'Anca et al., 2016).

In this perspective, the CDAM was designed and developed in order to hide the internal complexity of the underlying DSS model and ease the submission of the execution and the retrieval of the results by the upper layers of the operational chain. Moreover, a high modularity implements the separation of concerns, while the adoption of standard interfaces and existing technologies, such as JSON format and REST web services, ensures flexibility and interoperability of the software components. Finally, in order to guarantee a high security level, the incoming requests rely on the HTTPS[22] protocol, while a dedicated private network channel has been set up for the submission of VISIR jobs on the target computing infrastructure.

From a technological viewpoint, a two-layer logical architecture has been implemented (see Fig. 2). The first layer, the CDAM-Gateway, has been designed to be the entry point for job submissions; it is responsible for managing the incoming requests and for interfacing with the target infrastructure for the algorithm execution. The second layer, the CDAM-Launcher, has the responsibility for performing the submission and correctly managing the job execution.

Delving into more detail on the CDAM-Gateway, it consists of two modules: the RESTful web interface and the CDAM-Scheduler. The former provides the SSA platform with a uniform REST interface for accepting and managing the job execution requests provided in a JSON format. The latter sets the DSS execution environment and forwards the correct input parameters to the CDAM-Launcher.

The CDAM-Launcher is hosted on the target computing infrastructure and is responsible for properly managing the job submission. It relies on the workload manager SLURM[23] to perform the execution in a cluster environment. As in the case of the CDAM-Scheduler, the launcher provides a specific module for the DSS submission management. From an operational point of view, an incoming request for an execution of the VISIR model triggers the following sequence of operations on the CDAM-Launcher side: (i) check the parameters received from the CDAM-Scheduler; (ii) prepare the input files needed by the model; (iii) launch the model execution; (iv) once the job is completed, contact the SSA platform; (v) send back the results.

Following the same approach taken for the job request, the sent-back results are first of all embedded into a JSON object compliant with the specific JSON schema defined for the CDAM.

4 Functioning of the operational system

Following the more structural information from Sect. 3, this section aims to document the functioning of the VISIR operational system at a higher level. It includes the execution logic of the system, the system performance, and the end-user interaction with the system.

4.1 Execution logic

For the description of the logical execution of VISIR, we hereafter refer to the numbered steps in the sequence diagram of Fig. 3.

The access to VISIR is open after authentication (1). At the moment, however, also a valid subscription is also required in order to run the route computation service. The authentication is then granted by the DSS portlet hosted on the SSA platform (2). At this point, after departure and arrival location for the route have been set, a route computation request can be submitted (3) in the form of a string of parameters (see Sect. 7). Once it is forwarded through the DSS portlet (3′) to the MB, a correlation ID is provided back to the DSS portlet (4a) and the client application (4b). At the same time, the MB forwards the request to the CDAM (4c). While the VISIR model is run on the supercomputing facility controlled by the CDAM (6a–c), a polling activity starts at the client (5a) and is forwarded down to the level of the MB (5a′). Such a querying is iterated until the model results from the

[22] https://tools.ietf.org/html/rfc2818

[23] http://slurm.schedmd.com/

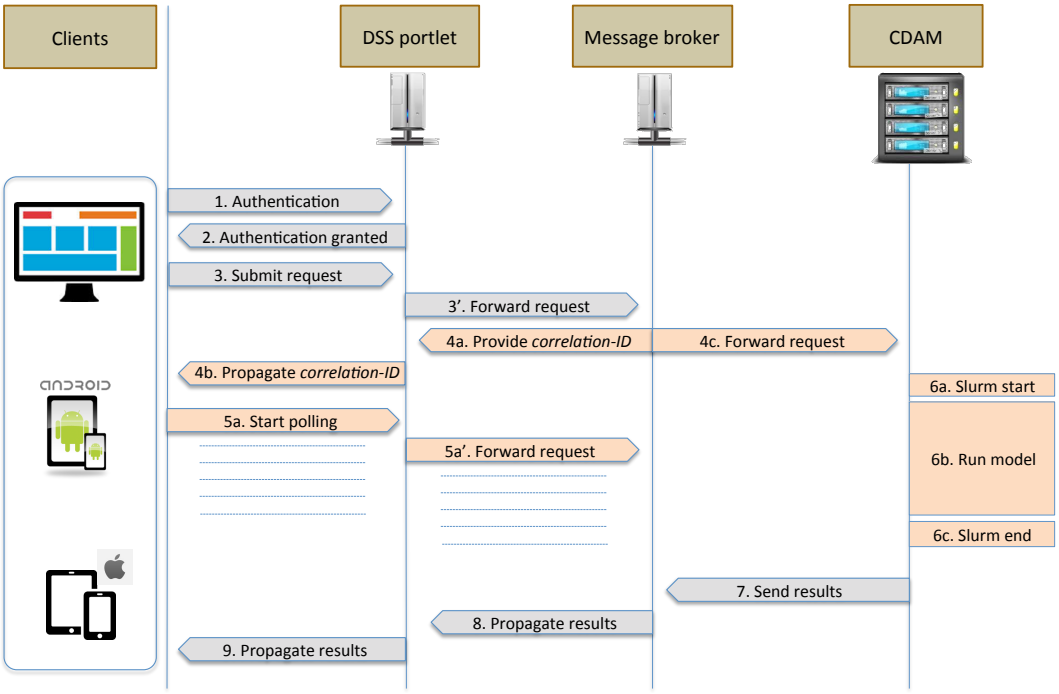

Figure 3. Data flux diagram. The downward-oriented vertical coordinate is the time elapsed since the user started interacting with VISIR, while the horizontal coordinate goes from high-level to low-level operations. Orange shading indicates simultaneously occurring operations. The dotted lines indicate replicas of steps (5a) and (5a′), performed until CDAM results are made available to the MB. The correlation-ID identifies the VISIR job, see route register in Fig. 7a and Sect. 7.

Table 2. Correspondence between steps within the data flux diagram (Fig. 3) and nodes of the VISIR system (cf. Fig. 4). The step part (s: start, e: end) is also given when relevant (– otherwise).

	U_0	U_1	S_1	C_1	m	C_2	S_2	p	U_2
Data flux diagram (Fig. 3) step #	1	5a	4c	6a	6b	6c	7	5a–8	9
Step part	s	s	s	s	–	e	e	s–e	e

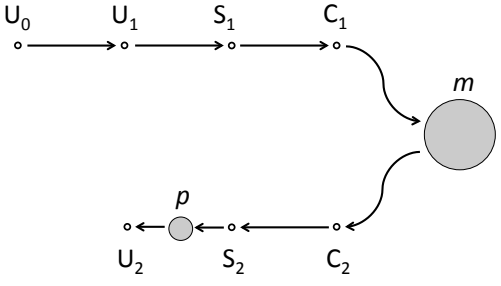

Figure 4. Graph structure of the TESSA system for running DSSs like VISIR. The nodes refer to points of time measurements: the user interface (U), the SSA platform (S), and the CDAM (C). The m (MATLAB job run) and the p (polling) nodes are displayed as larger filled circles for highlighting the fact that they contain the internal waiting times $\tau(m)$ and $\tau(p)$ respectively.

CDAM are sent to the MB (7). At that time, they are immediately propagated up to the level of the DSS portlet (8) and, upon a polling mechanism (Eq. 1), to the client (9).

The different logical functions played by the clients, the DSS portlet, the MB, and the CDAM, enable a physical separation of the assets of the system. This is indeed the case, since the TESSA architecture mirrors the organizational structure of the TESSA consortium.

4.2 System performance

As outlined in the previous subsections, the TESSA architecture consists of various connected components acting asynchronously.

In order to assess its performance, it is convenient to consider it as a network whose nodes are linked in a directional way, as in Fig. 4. The network is characterized by delays along the links and by node internal waiting times. Some logical steps of the flux diagram of Fig. 3 can be remapped

Table 3. Parameters and scores for the least-square fits shown in Fig. 5. For the η and τ signals (a, c panels of Fig. 5) all fits are of type $ax^b + c$, while for the δ signals (b, d panels of Fig. 5) the fits are of type $dx + e$. The b exponent is given with the 95 % uncertainty bounds, while the e intercept is given with the estimated variance (among brackets). Least-χ^2 linear fits (Fig. 5b, d) employ formulas provided in Flannery et al. (1992, chap. 15) for the coefficients of the linear regression and their uncertainties. M = nautical mile = 1850 m; kB = kilobytes.

			Units	Motorboat	Sailboat
UI	η	a	$\mathrm{s\,M^{-}}^{b}$	3.9×10^{-7}	2.4×10^{-5}
		b	–	3.3 (2.6–3.9)	3.0 (2.5–3.5)
		c	s	19.8	14.1
		R^2	%	98.5	99.5
	δ	d	$\mathrm{kB\,s^{-1}}$	90 (100)	30 (18)
		e	s	7.1 (1.2)	6.3 (0.7)
		R^2	%	80.6	59.3
SSA	η	a	$\mathrm{s\,M^{-}}^{b}$	9.3×10^{-7}	1.8×10^{-6}
		b	–	3.1 (2.6–3.7)	3.0 (2.5–3.5)
		c	s	14.5	10.5
		R^2	%	98.3	99.6
	δ	e	s	3.2(0.2)	3.1(0.2)
		R^2	%	74.1	94.2
CDAM	η	a	$\mathrm{s\,M^{-}}^{b}$	1.1×10^{-6}	2.0×10^{-6}
		b	–	3.1 (2.6–3.7)	3.0 (2.5–3.5)
		c	s	12.6	8.6
		R^2	%	99.6	99.6
	δ	e	s	2.1 (0.1)	2.1 (0.1)
		R^2	%	98.7	94.8
MATLAB	τ	a	$\mathrm{s\,M^{-}}^{b}$	1.7×10^{-6}	3.7×10^{-6}
		b	–	3.0 (2.5–3.6)	2.9 (2.4–3.4)
		c	s	9.3	5.5
		R^2	%	97.9	99.5

to the network of Fig. 4 via the correspondence provided in Table 2. The nodes of Fig. 4 were chosen because they correspond to the places where it is presently possible to carry out time measurements. The names of the nodes, $\{U, S, C\}$, correspond to the names of the TESSA tiers (UI, SSA platform, CDAM) where measurements are taken. The performance of the VISIR operational system can then be assessed by comparing time measurements at various nodes.

The experimental activity was carried out ensuring that the computational cluster was nearly idle. Computations of routes up to 500 M length were requested from the UI. For each given route length, the same departure and arrival were employed in up to 10 routing jobs during test sessions occurring on three different calendar dates. The experimental protocol also included recording timestamps at the various nodes of the network of Fig. 4.

In Fig. 5 we display data collected for both motorboat (Fig. 5a, b) and sailboat (Fig. 5c, d) VISIR jobs. In the left panels we display the "waiting times" $\eta(Q) = t(Q_2) - t(Q_1)$, with $Q = \{U, S, C\}$, and $\tau(m)$, the duration of the MATLAB job. The measurement of the $\eta(Q)$ times, which are differences of absolute times at a given tier of the TESSA infrastructure, does not require clock synchronization among the asynchronous system tiers.

For longer routes, $\eta(Q)$ is dominated by $\tau(m)$, which with respect to route length L, scales as a power law with an exponent of about 3 (cf. Table 3). The relation between L and the number N of grid points in the selected bounding box for route computation (see Sect. 4.3) depends on the aspect ratio of the latter. For a squared box, $L \sim \sqrt{N}$, for a more elongated one, $L \sim N$. Consequently, the fitted exponent also implies a power-law dependence on N with an exponent comprised between 1.5 and 3. This is compliant with the fact that a quadratic trend with respect to N was fitted in Mannarini et al. (2016b). This in turn agrees with the theoretical performance of Dijkstra's algorithm (Sect. 2.2) for its implementation without any specific data structures. Thus, the performance of the VISIR operational system for long routes mirrors that of the model for ship route optimization.

For shorter routes however, it is apparent that the values of $\eta(Q)$ at the various system layers differ by several seconds. In order to better explore the performance of the VISIR operational system, we subtract the $\tau(m)$ dominant contribution, defining the excess times $\delta(Q) = \eta(Q) - \tau(m)$. Furthermore, we display $\delta(Q)$ vs. the sum of the sizes of the two JSON files containing the geodetic and the optimal route (see Sect. 7). That is a proxy of the size of the payload transferred from the CDAM up to the level of the UI. Resulting data for the motorboat and sailboat modality are displayed in Fig. 5b, d.

Starting from the CDAM δ data (blue markers and lines), a nearly constant trend is observed for both motor- and sailboat tests. We find $\delta(C) \approx 2$ s. Between the C_2 and C_1 nodes two distinct processes occur on the CDAM: the MATLAB job is submitted via slurm and, upon completion, the results are uploaded using the call-back URL mechanism (cf. Sect. 3.2) for making them available to the SSA platform. We separately tested that the duration of a slurm task that just submits a void MATLAB job is between 1.4 and 2.0 s. Thus, most of the $\delta(C)$ time should be ascribed to this irreducible slurm duration. Also, $\delta(C)$ is found to be independent of payload size, as a consequence of the high-speed network connection of the computing facility hosting the CDAM.

For the SSA δ data (red markers and lines), again a nearly constant trend is observed for both motor- and sailboat job tests. We find $\delta(S) \approx 3$ s. Despite the name (arising from the TESSA tier where the measurements are carried out), the $\delta(S)$ duration contains, besides the SSA–CDAM bidirectional delays, mainly CDAM tasks: the preliminary validation of the job request parameters (occurring at the CDAM-Gateway), the setup of the remote environment on the supercomputing facility (CDAM-Gateway), the namelist production (CDAM-Launcher), the processes leading to $\delta(C)$ (CDAM-Launcher), and the creation of an output JSON file

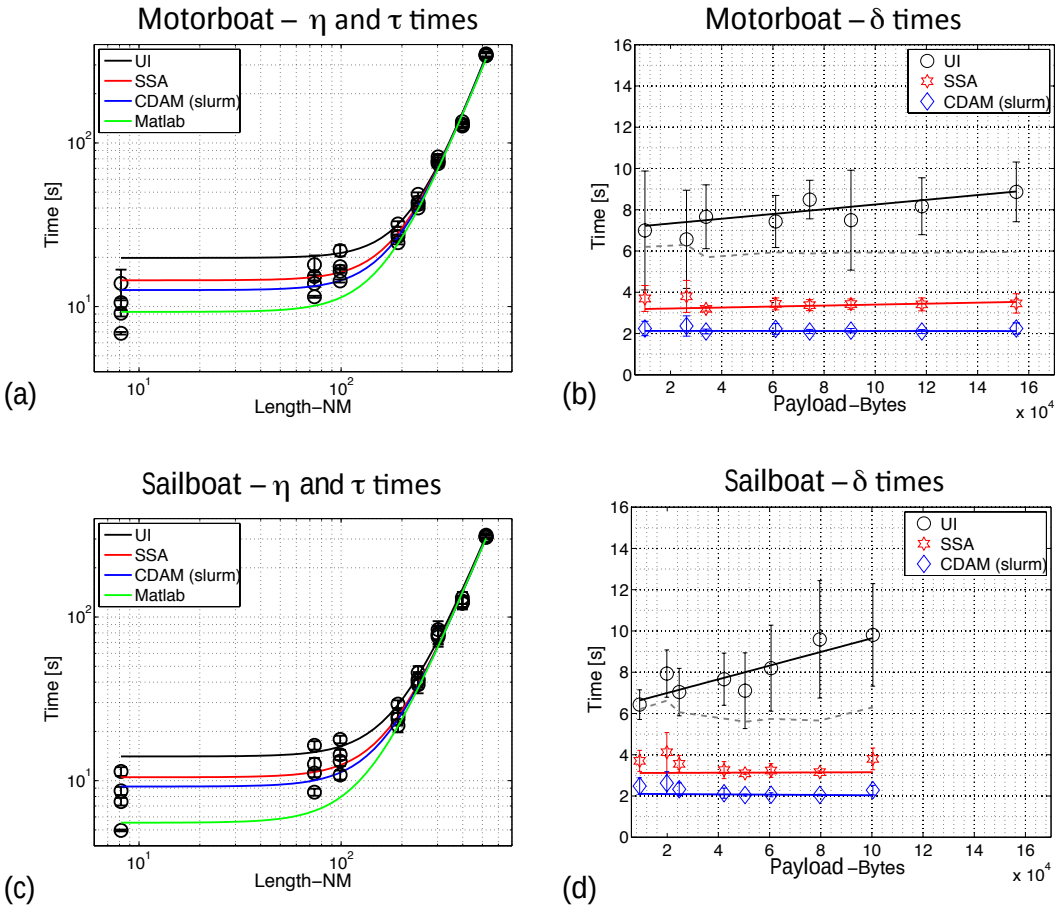

Figure 5. Performance of VISIR operational system, distinguishing between motorboat **(a, b)** and sailboat modality **(c, d)**. Left panels contain performance vs. route length L. The "UI", "SSA", and "CDAM (slurm)" times are respectively the quantities $\eta(U)$, $\eta(S)$, and $\eta(C)$, while "MATLAB" stands for $\tau(m)$. Power-law least-square fits are displayed as solid lines, and fit parameters are given in Table 3. Right panels contain the $\delta(Q) = \eta(Q) - \tau(m)$ times. Here linear fits are displayed as solid lines and the times $\delta(S) + t_{\mathrm{poll}}/2$ as dashed grey lines. The length of the vertical bar is given by $2\sigma_{\mathrm{e}}$, where the estimated standard deviation based on a sample of ν measurements is computed as $\sigma_{\mathrm{e}} = \sqrt{\sum_i (y_i - \overline{y})^2/(\nu - 1)}$.

containing both the geodetic and the optimal route. With the exception of the first and the last, these processes are independent of payload size and, thanks to the fair internet bandwidth available at the SSA platform location, result in a nearly constant $\delta(S) - \delta(C) \approx 1$ s.

Moving to the UI δ data (black markers and lines), we note first of all that, for a given route length, the sailboat's payload is smaller than the motorboat's. This is due to the fact that a smaller number of fields is stored in the sailboat payload (e.g. the flags of the dynamical safety constraints, see Sect. 2, are just available for the motorboat case). Furthermore, for both motorboat and sailboat, the UI measurements are characterized by a large variability (error bar size) among the different tests. The data show a linear trend of $\delta(U)$ with respect to payload size. The intercept of the linear fits (e parameter in Table 3) is located at about 7 s, and $\delta(U)$ is found to increase with payload size.

In order to understand the results about the e offset, it should be recalled that $\delta(U) - \delta(S)$ includes, besides the UI–SSA bidirectional delays, the waiting time $\tau(p)$ due to the polling interval t_{poll} of the client application (Sect. 3.2). The polling mechanism can be considered as a normally closed gate, opening instantaneously any t_{poll} time units. Thus, a signal arriving at the gate at a time σ cannot pass before the additional delay $\tau(p)$, given by

$$\tau(p) = t_{\mathrm{poll}} - \mathrm{mod}(\sigma, t_{\mathrm{poll}}), \tag{1}$$

is elapsed. Eq. (1) defines a piece-wise linear function of σ. In the case of VISIR, σ includes, as described above, both deterministic and stochastic (due to internet bandwidth and computing resources availability) processes. Thus, it is convenient to estimate the time average of $\tau(p)$, given by

$$\overline{\tau(p)} = \langle \tau(p) \rangle_{\sigma} = \frac{1}{2} t_{\mathrm{poll}}. \tag{2}$$

Thus, we find that, on the average, it must be $\delta(U) - \delta(S) \geq t_{\mathrm{poll}}/2$. For this reason, we display in Fig. 5b, d the quantity $\delta(S) + t_{\mathrm{poll}}/2$ (dashed grey line), and as expected we find that $\delta(U)$ is never smaller than it. At present, the setting of the polling mechanism is such that $t_{\mathrm{poll}} = 5$ s.

Concerning the slope found in Fig. 5b, d for the linear fits of $\delta(U)$, we note that it is different for the motorboat and sailboat modality (Table 3). We recall that the δ signals displayed are not affected by the VISIR model internal time $\tau(m)$, which in fact differ between motorboat and sailboat computations, as shown by the values in Table 3. Thus, the reason for the different slopes must be searched in either the internet connection or some UI processing. The error bars in Fig. 5b, d and the uncertainty in the fitted parameters (Table 3) confirm a large variability of the internet speed during the tests. However, there are also specific data checks and rendering peculiarities in the line chart (Fig. 8b, c) of the VISIR sailboat interface. Thus, at the moment we cannot rule out any of the possible explanations. Both are likely to be at play, though more accurate tests and measurements would be needed for assessing their relative importance.

Ultimate performance limit of VISIR

The experimental findings above and their analysis enable us to assess the ultimate performance limit of the VISIR technological infrastructure.

There are limiting factors of two types: on the one hand, there are parameters (such as t_{poll} and the slurm duration for submitting a job) that could be tuned for obtaining some minor improvement (a few seconds in the best case). On the other hand, there are larger penalties paid to the present architecture which represent the actual bottlenecks of performance:

i. the computational cost for generating the ship routes

ii. the internet-based communication among asynchronous system components.

In particular, (i) dominates the UI waiting time for long routes while (ii) is the limiting factor for the shorter routes. Fixing item (i) corresponds to a situation where dedicated computational resources are available and the performance of the computing algorithm has significantly improved. It can be simulated by subtracting the duration $\tau(m)$ of the MATLAB job, as we did in Fig. 5b, d.

If further also item (ii) were fixed by a fast link (optical fiber or so), the ultimate limit for the VISIR user waiting time would be given by $e + t_{\mathrm{poll}}/2$, where e is the intercept of $\delta(S)$. Using current values for both the e coefficient (cf. Table 3) and t_{poll}, this would imply a total waiting time of about 6 s after a route request is submitted from the user interface.

4.3 Use of the system

The end-user experience of VISIR entirely occurs within the presentation layer of the application stack. Its main elements are a menu, a geographic map, and a line chart[24]. Upon authentication, the submission of a route computation can be completed by a minimum of three clicks. The left menu offers three modalities to enter the two locations between which the route has to be computed: (i) by typing their lat–long coordinates; (ii) by clicking–tapping on their positions on the map; (iii) by typing the name of a land-based location. The latter option exploits, depending on the platform, a Google or MapKit Framework API for georeferencing toponyms (e.g. "Otranto" is mapped to 40.14390° N, 18.49117° E).

A crucial point is subsetting the domain ("bounding box") used by the algorithm for computing the optimal route. In fact, the computational cost of the route computation heavily depends on the bounding box extent. By our choice, it initially corresponds to a box whereby two opposite vertices are the route endpoints. However, due to specific domain topology (peninsular or archipelagic regions) and environmental conditions (leading to unsafe navigation), a suboptimal route or no route at all may result from using the default bounding box. Thus, we designed all the interfaces in a way that the bounding box can be interactively resized. Leaving it to the end-user to decide whether and how to resize the bounding box certainly introduces a degree of subjectivity in the final results of the route computation. However, it is our experience that, after a few trials, the user easily learns how to do it in a conservative and still effective way. In fact, it is sufficient to obtain a result and then submit a new route with a larger bounding box; if there is no change in the results, convergence has been achieved.

Furthermore, if a land-based endpoint is selected (such as "Lecce"), the next sea position within the bounding box and with positive UKC (i.e. sea depth larger than vessel draught) is automatically retrieved by the model. It is also possible to set a minimum offshore distance ρ to be met by each route waypoint (see Fig. 6). This is achieved by masking the graph domain previously to the run of the shortest-path algorithm (Mannarini et al., 2016b).

Three pre-defined sets of motorboat parameters (a displacement hull cabin cruiser, a fishing vessel, and a small ferryboat) can be selected and edited. Alternatively, a sailboat type can be picked up from a database of boats with lengths between 7 and 19 m (Fig. 8a). From the "advanced settings" section of the left menu (in case of motorboat) the individual safety constraints for motorboat routing can be checked and the voluntary speed reduction can be chosen as an option.

[24] At present, the line chart is just available for the web application (v.4.2). It is planned that this feature will be made available for the mobile apps, too.

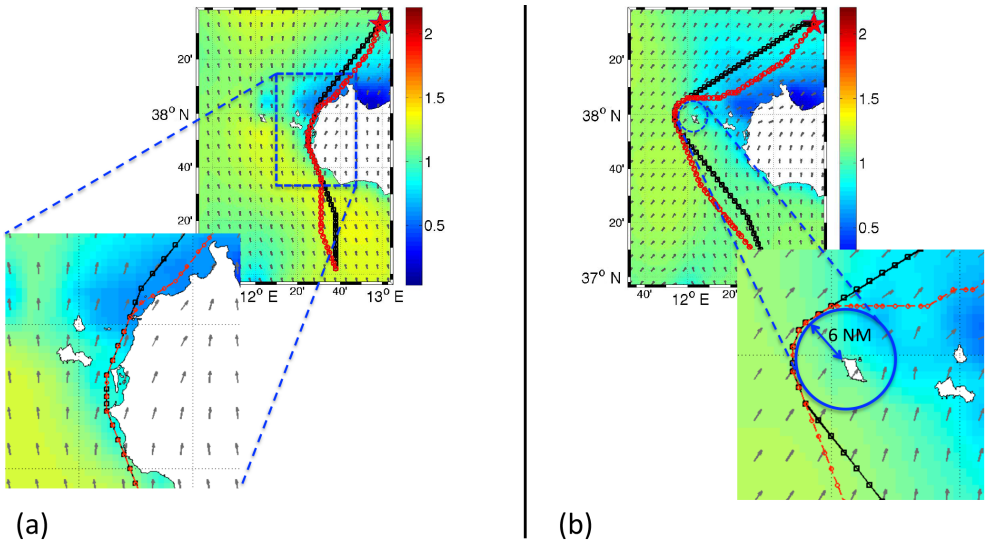

Figure 6. Routes with minimum offshore distance ρ set to zero (panel **a**) or to 6 M (panel **b**). The inset of **(a)** displays that even for $\rho = 0$ the land mass is avoided, while the inset of **(b)** demonstrates that the routes are locally tangent to a circle of radius $\rho = 6$ M centred on the island of Marettimo, off the western coast of Sicily.

Upon route submission, a progress bar with an estimate of the waiting time appears. Due to features of the algorithm, Sect. 2.2, such a time depends on both the bounding box size and the duration of the navigation. The random availability of the internet band and the unpredictable load of the computing resources do not allow the end-user to be given an exact estimation of such waiting time (Sect. 4.2). This is why the VISIR dialog window provides the time together with an uncertainty, whose value is identified by means of empirical tests, like those provided in Fig. 5.

If the route computation fails, a diagnostic message appears with a hint to what should be changed in order to get a successful result. If the route computation succeeds, both a geodetic (black) and an optimal route (red) are displayed on the map.

5 Examples

In order to demonstrate the operational functioning of VISIR, we report in this section about two execution scenarios, one for motorboat and the other one for sailboat.

5.1 Motorboat route

In Fig. 7 a fishing vessel route in the Aegean Sea is displayed. The geodetic route connects departure and arrival location while skipping the islands in between. The optimal route, besides avoiding the shoreline, contains a significant eastbound diversion (Fig. 7a). This is instrumental in avoiding the rough seas experienced along the geodetic route. The resulting optimal route, though 24 M longer, is nearly 1 h faster than the geodetic one.

The VISIR web application displays various combinations of forecast fields on the geographic map. In Fig. 7a the peak wave period and wave direction fields are shown. Furthermore, the line chart below the map can display various kinematic and environmental fields along the routes (Fig. 7b, c). Each waypoint (WP) on the map is bijectively linked to the corresponding WP on the line chart (see Fig. 7a, b), enabling an immediate georeferencing of line chart information, or, inversely, temporal localization of a WP. Furthermore, upon querying a WP (either from the map or from the line chart), the map of the forecast field at that specific time step is displayed, highlighting the time-dependent information processed by the VISIR model. In Fig. 7b the vessel speed over ground (SOG) for both routes is compared to the top speed of the vessel in calm weather conditions. It is apparent that the optimal route, involving sailing in calmer seas, enables the vessel's propulsion system to sustain a larger SOG. The route performance information is summarized in the route register in the left menu (Fig. 7a). As another option, the safety indices for vessel stability discussed in Mannarini et al. (2016b) can be displayed in the line chart. For instance, Fig. 7c shows that, for the selected route, pure loss of stability is experienced along the geodetic route, starting from positions at 110 M from the departure place. The optimal route instead per construction does not suffer from this hazard. However, it is up to the user to individually uncheck the default flags of the safety constraints: parametric rolling, pure loss of stability, surfriding and broaching.

In Fig. 7 the same route on top of the significant wave height and wave direction fields is shown, as it is visualized on an iPad (Fig. 7d) and an iPhone (Fig. 7e) screen. The Map-

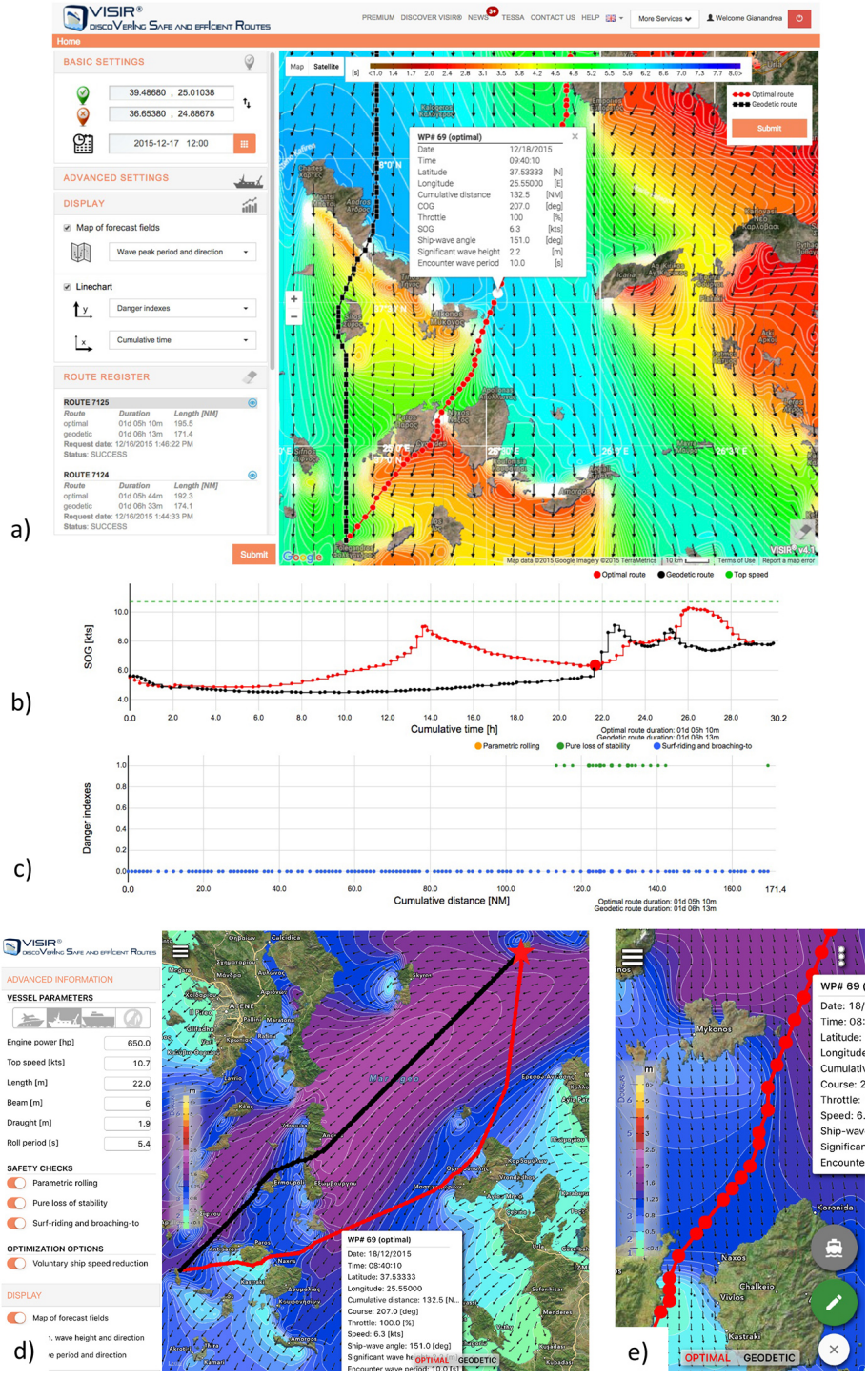

Figure 7. A motorboat route computed by VISIR. A displacement vessel with parameters as in Table 1 is employed for the computations. Panels **(a)**, **(b)**, and **(c)** belong to the web application; panels **(d)** and **(e)** to the iOS mobile app respectively on an iPad and iPhone. The shaded field in **(a)** is the peak wave period (see footnote no. 8 on p. 3) and wave direction, while the shaded field in **(d)** and **(e)** is significant wave height and wave direction. Panels **(b)** and **(c)** are two possible visualizations of the line chart below the map of the web application: in **(b)** the SOG vs. cumulative time since departure and in **(c)** the danger index vs. cumulative distance from departure place is visualized. See Sect. 7 for the availability of input and output data relative to this route.

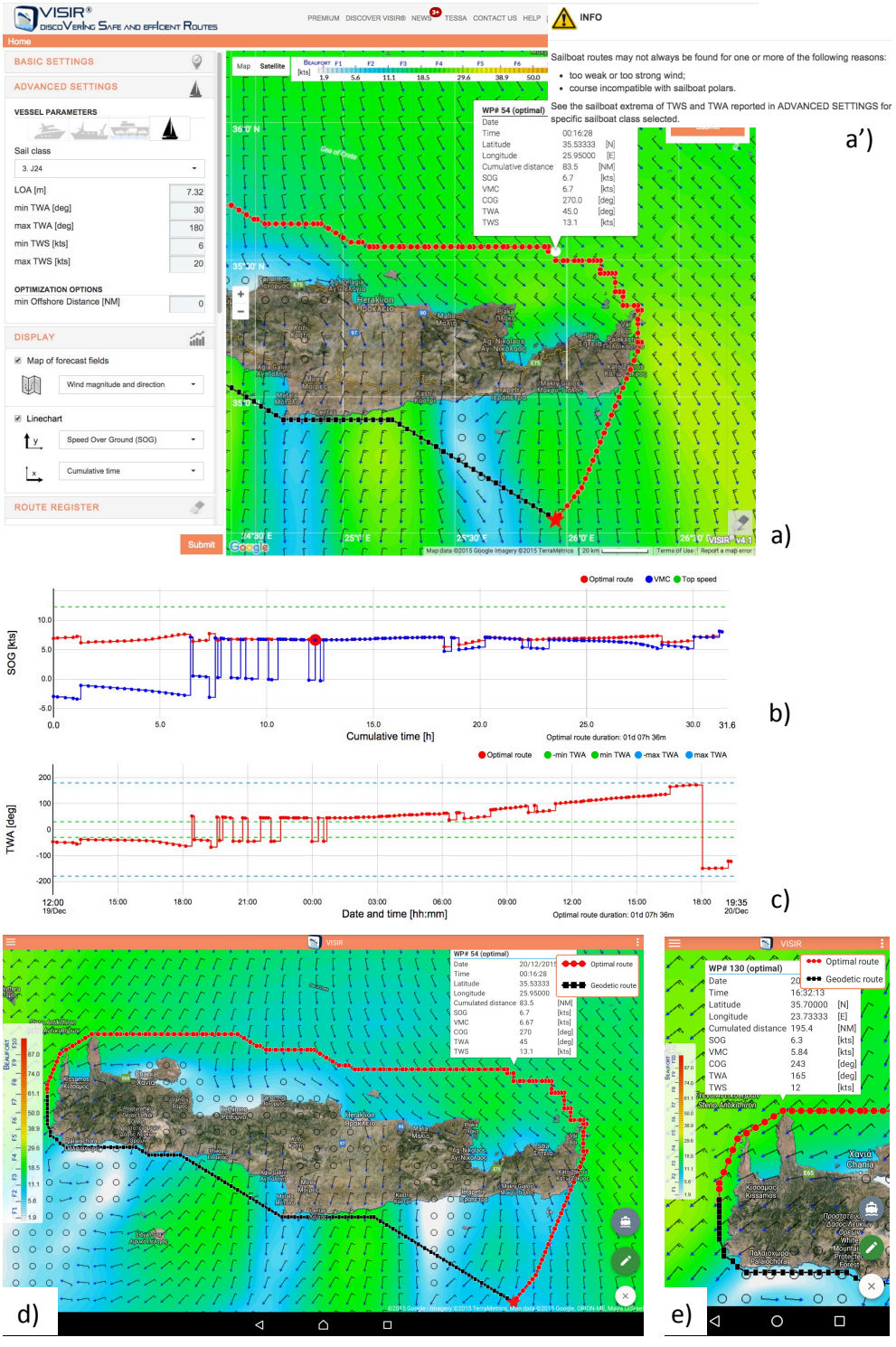

Figure 8. A sailboat route computed by VISIR. The shaded field in the maps is the 10 m height wind intensity and its direction. A "First 36.7" sailboat is employed for the computations. Panels **(a)**, **(a')**, **(b)**, **(c)** are relative to the web application; panels **(d)** and **(e)** to the Android mobile app on a tablet and a smartphone respectively. Panels **(b)** and **(c)** are two possible visualizations of the line chart below the map of the web application: in **(b)** the SOG vs. cumulative time since departure and in **(c)** the TWA vs. date and time (note the no-go-zone, Sect. 2.3, between the green dashed lines) is visualized. See Sect. 7 for the availability of input and output data relative to this route.

Kit Framework cartography and the possibility to rotate the geographical north of the map can be noted.

5.2 Sailboat route

First of all, we should stress the fact that, due to the limited capability of producing thrust through wind, a sailboat route between given endpoints is not always feasible. In particular, there are limitations for upwind motion and for too weak or too strong wind intensities (Mannarini et al., 2015). For this reason, it is even less likely that the most direct route between the endpoints is feasible. Thus, in VISIR-I the sailboat geodetic route is provided just as a topological information, without any reference to its kinematical aspects. Furthermore, the end-user is informed about the possible difficulty in computing the route by an info box including a hint to check the polar plot parameters (Fig. 8a′).

In Fig. 8, a First 36.7 (an 11 m long boat) route around the island of Crete is displayed. In the actual case shown, it is seen that the geodetic and the optimal route sail on opposite sides of the island (Fig. 8a, d, e). This is suggested by the algorithm in order to avoid the dead calm areas in the lee of the high (> 2000 m a.s.l.) mountains of Crete. The line charts selected for this case study show the SOG and the true wind angle (TWA) with respect to cumulative time and date/time since departure, respectively. The SOG line chart (Fig. 8b) also includes the velocity made good to course (VMC) information (Mannarini et al., 2015), demonstrating that the initial eastbound diversion of the optimal route locally implies departing from the target ($\mathrm{VMC} < 0$). The TWA line chart (Fig. 8c) shows that the algorithm suggests the boat sails at a TWA close to the minimum possible for the actual polar plot during the close-hauled phase (upwind) and at a TWA close to the maximum possible during the broad reach phase (downwind).

The sailboat routing is at the moment available, besides on the web application, just on a development version of the Android mobile application. A few previews of the tablet (Fig. 8d) and smartphone (Fig. 8e) layouts are displayed in this paper.

6 Conclusions

Starting from the VISIR model, a DSS for on-demand computation of optimal ship routes in the Mediterranean Sea has been put into operations. It addresses small displacement hull motorboats and sailboats of various sizes. Shoreline and bathymetry are the static databases used for checking the minimal requirements for the safety of navigation. Forecasts of sea state and wind from third-party providers are the dynamical information used for checking vessel stability and for the minimization of the route duration. The operational system is available cross-channel. Client applications for the web browser and iOS and Android devices have been developed.

The infrastructure developed for running VISIR in an operational way is to a large extent general, and it was successfully employed as a template for other DSSs developed within the TESSA project (Liubartseva et al., 2016 and Coppini et al., 2016a).

One of the most significant authors' experience is that the realization of the VISIR operational system (2013–2015) also had a positive feedback on the initial phase (2012–2014) of development of the VISIR ship routing model. This was due to the fact that the operational service required the model to reach a level of robustness far beyond the case study testing. Exceptional and challenging environmental conditions highlighted issues and bugs of the model, and the demanding computational requirements of a multiuser system pushed for a more and more efficient model code.

It is important to stress that the technological infrastructure developed for running VISIR in operational mode allows an end-user to drive the execution of a state-of-the-art research model, customizing his/her route needs, vessel features, and level of safety. The fact that the scientific and computational issues behind the UI are made transparent to the UI end-user is one of the main achievements of the architecture. Furthermore, it allows for avoiding heavy computations on the device used to access the system while minimizing the amount of data transferred to it from the computational facility. The performance of the TESSA architecture VISIR runs within has been assessed through extensive experimental tests. They indicate that, upon removal of the major bottlenecks, the ultimate limit for the VISIR end-user waiting time could be a few seconds (Sect. 4.2).

Numerous developments of the VISIR ship routing model are envisioned, and some of them were already discussed in Mannarini et al. (2016b). As a development possibility for the operational infrastructure, we envision the realization of a "VISIR web API" for granting interoperability with third-party softwares. Furthermore, interoperability along the lines of the new STM concept is also possible. In particular, the routes computed by VISIR can be made compliant with the route exchange format RTZ recently approved by IEC (International Electrotechnical Commission) as an international standard (see footnote no. 4 on p. 2).

7 Data availability

Both the input and the output data relative to the case studies displayed in Figs. 7 and 8 are available in the Supplement of this manuscript.

In the input data directory we provide:

- The submission string generated upon user request of a route computation (Sect. 4.1);

– The namelists (`*_pars.txt` files) for running VISIR Mannarini et al. (2016b, a), prepared by the CDAM-Launcher out of the submission string;

– The `algo.out` file, which contains the MATLAB command window output during job execution.

In the output data directory we provide:

– A log of the input namelist (`2_namelist_log.txt`) that can be used for checking correspondence to the input parameters;

– The detailed voyage plans relative to both the geodetic (`2_gdt_*`) and optimal (`2_opt_*`) routes, in both a tabular (`.tab`) and JSON (`.json`) format (cp. Sect. 4.2). Note that The WP key in the `2_voyageplan.*` files corresponds to the WP value in the tool-tips shown in Figs. 7 and 8;

– A report of the main results from the VISIR job (`2_log.txt`).

The Supplement related to this article is available online at doi:10.5194/nhess-16-1791-2016-supplement.

Acknowledgements. Funding through TESSA (PON01_02823/2) project is gratefully acknowledged. Yogesh Kumkar and Andrea Villani are thanked for their initial contribution to the development of the operational infrastructure and the VISIR mobile application for Android, respectively. Roberto Bonarelli provided appreciated feedback on the sailboat interface. Charlotte P. Martinkus contributed to the experimental activity to assess the system performance.

Edited by: A. Olita
Reviewed by: C. Granell and one anonymous referee

References

Belenky, V., Bassler, C. G., and Spyrou, K. J.: Development of Second Generation Intact Stability Criteria, Tech. rep., DTIC Document, 2011.

Bowditch, N.: The American Practical Navigator: An Epitome Of Navigation (Bicentennial edition), NIMA Pub, Bethesda, Maryland, 2002.

Coppini, G., Jansen, E., Turrisi, G., Creti, S., Shchekinova, E. Y., Pinardi, N., Lecci, R., Carluccio, I., Kumkar, Y. V., D'Anca, A., Mannnarini, G., Martinelli, S., Marra, P., Capodiferro, T., and Gismondi, T.: A new search-and-rescue service in the Mediterranean Sea: a demonstration of the operational capability and an evaluation of its performance using real case scenarios, Nat. Hazards Earth Syst. Sci. Discuss., doi:10.5194/nhess-2016-175, in review, 2016a.

Coppini, G., Marra, P., Lecci, R., Pinardi, N., Cretì, S., Scalas, M., Tedesco, L., D'Anca, A., Fazioli, L., Olita, A., Turrisi, G., Palazzo, C., Aloisio, G., Fiore, S., Bonaduce, A., Kumkar, Y., Ciliberti, S. A., Federico, I., Mannarini, G., Agostini, P., Bonarelli, R., Martinelli, S., Verri, G., Lusito, L., Rollo, D., Cavallo, A., Tumolo, A., Monacizzo, T., Spagnulo, M., Sorgente, R., Cucco, A., Quattrocchi, G., Tonani, M., Drudi, M., Panzera, L., Navarra, A., and Negro, G.: SeaConditions: a web and mobile service for safer professional and recreational activities in the Mediterranean Sea, Nat. Hazards Earth Syst. Sci. Discuss., doi:10.5194/nhess-2016-176, in review, 2016b.

D'Anca, A., Conte, L., Nassisi, P., Palazzo, C., Lecci, R., Cretì, S., Mancini, M., Nuzzo, A., Mirto, M., Mannarini, G., Coppini, G., Fiore, S., and Aloisio, G.: A multi-service data management platform for scientific oceanographic products, Nat. Hazards Earth Syst. Sci. Discuss., doi:10.5194/nhess-2016-177, in review, 2016.

de Jong, P., Katgert, M., and Keuning, L.: The development of a Velocity Prediction Program for traditional Dutch sailing vessels of the type Skûtsje, in: 20th HISWA Symposium, 2008.

Dijkstra, E. W.: A note on two problems in connexion with graphs, Numerische mathematik, 1.1, 269–271, 1959.

Fielding, R. T.: Architectural styles and the design of network-based software architectures, Ph.D. thesis, University of California, Irvine, 2000.

Fiorini, M. and Lin, J.-C. (Eds.): Clean Mobility and Intelligent Transport Systems, Transportation Series 1, IET, 187–217, 2015.

Flannery, B. P., Press, W., Teukolsky, S., and Vetterling, W. T.: Numerical Recipes in FORTRAN 77: The Art of Scientific Computing, Cambridge University Press, New York, 650–660, 1992.

Hey, A. J., Tansley, S., and Tolle, K. M. (Eds.): The fourth paradigm: data-intensive scientific discovery, vol. 1, Microsoft research Redmond, WA, 13–19, 2009.

IMO: MSC. 1/Circ. 1228 Revised guidance to the Master for avoiding dangerous situations in adverse weather and sea conditions, International Maritime Organization (IMO), London, UK, 1–6, 2007.

Jansen, E., Coppini, G., and Pinardi, N.: Drift simulation of MH370 debris using supersensemble techniques, Nat. Hazards Earth Syst. Sci., 16, 1623–1628, doi:10.5194/nhess-16-1623-2016, 2016.

Kjølleberg, J.: Weather Routing of Supply Vessels in the North Sea-Solving the Supply Vessel Weather Routing Problem, Master's thesis, NTNU Trondheim, 2015.

Levin, M.: Designing Multi-device Experiences: An Ecosystem Approach to User Experiences Across Devices, O'Reilly Media, Inc., 1–130, 2014.

Liubartseva, S., Coppini, G., Pinardi, N., De Dominicis, M., Lecci, R., Turrisi, G., Creti, S., Martinelli, S., Agostini, P., Marra, P., and Palermo, F.: Decision support system for emergency management of oil spill accidents in the Mediterranean Sea, Nat. Hazards Earth Syst. Sci. Discuss., doi:10.5194/nhess-2016-174, in review, 2016.

Lu, R., Turan, O., Boulougouris, E., Banks, C., and Incecik, A.: A semi-empirical ship operational performance prediction model for voyage optimization towards energy efficient shipping, Ocean Eng., 110, 18–28, 2015.

Mannarini, G., Lecci, R., and Coppini, G.: Introducing sailboats into ship routing system VISIR, in: Information, Intelligence,

Systems and Applications (IISA), 6th International Conference on Information, Intelligence, Systems and Applications (IISA), IEEE, 1–6, doi:10.1109/IISA.2015.7387962, 2015.

Mannarini, G., Pinardi, N., and Coppini, G.: VISIR: A Free and Open-Source Model for Ship Route optimization, in: COMPIT 2016, edited by: Bertram, V., 161–171, Technische Universität Hamburg-Harburg, Hamburg, 2016a.

Mannarini, G., Pinardi, N., Coppini, G., Oddo, P., and Iafrati, A.: VISIR-I: small vessels – least-time nautical routes using wave forecasts, Geosci. Model Dev., 9, 1597–1625, doi:10.5194/gmd-9-1597-2016, 2016b.

Montes, A. A.: Network shortest path application for optimum track ship routing, Ph.D. thesis, Naval Postgraduate School, Monterey, California, 2005.

Orda, A. and Rom, R.: Shortest-path and Minimum-delay Algorithms in Networks with Time-dependent Edge-length, J. ACM, 37, 607–625, 1990.

Proctor, R. and Howarth, M.: Coastal Observatories and operational oceanography: a European perspective, Mar. Technol. Soc. J., 42, 10–13, 2008.

Scalas, M., Marra, P., Tedesco, L., Quarta, R., Cantoro, E., Tumolo, A., Rollo, D., and Spagnulo, M.: TESSA: design and implementation of a platform for Situational Sea Awareness, Nat. Hazards Earth Syst. Sci. Discuss., doi:10.5194/nhess-2016-166, in review, 2016.

Siwe, U., Lind, M., Hägg, M., Dalén, A., Brödje, A., Watson, R., Haraldson, S., and Holmberg, P.-E.: Sea Traffic Management – Concepts and Components, in: 14th International Conference on Computer and IT Applications in the Maritime Industries, Ulrichshusen, 11–13 May 2015, edited by: Volker, B., 281–289, Technische Universität Hamburg-Harburg, 2015.

Tolman, H. L.: User manual and system documentation of WAVEWATCH III TM version 3.14, technical note, MMAB Contribution, 2009.

Walther, L., Burmeister, H.-C., and Bruhn, W.: Safe and Efficient Autonomous Navigation with Regards to Weather, in: 13th International Conference on Computer and IT Applications in the Maritime Industries, Redworth, 12–14 May 2014, edited by: Volker, B., 303–317, Technische Universität Hamburg-Harburg, 2014.

Weintrit, A., Lee, S., and Alexander, L.: Software Quality Assurance Issues Related to e-Navigation, in: Marine Navigation and Safety of Sea Transportation: Advances in Marine Navigation, 73–77, CRC Press, London, UK, 2013.