How did the Ever Given get stuck in the Suez Canal?

Miguel De Le Court

June 2021

Abstract

This reports aims at studying the 2021 Suez canal incident with a 2d CFD model. We start by reconstructing the historical events using a simplified geometry for the canal and the ship. We then formulate an ALE FEM model where the mesh follows the boat in space, and use it to compute the forces on the ship. We then use this data to reproduce the events without enforcing the movements a priori. Based on a slightly tweaked simulation, we show that the incident likely could have been avoided if the ship had been controlled slightly differently. We also point out at the difficulty of finding such ship controls, due to some positive feedback loops that happen when steering the ship.

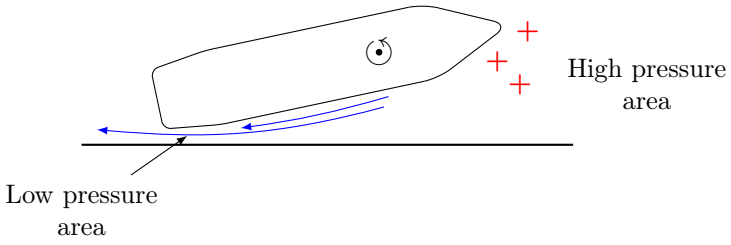
Contents

1 Introduction

1.1 Background and Research question

In March 2021, the Suez Canal was blocked for six days after the Ever Given container ship got stuck sideways in the canal. The Suez Canal is a route to about 12% of global trade, and it is estimated that the blockade cost around $9 Billion per day[1][2][3]. To this day, the effects of the incident are still affecting some parts of the economy[4][5], thus, understanding the causes of this incident may be of crucial importance.

One known effect that played a significant role in the incident, and which is the most common explanation for it, is the so-called bank effect. When a ship moves forward, an area of high pressure is created around the bow of the ship. The water that flows along its sides will, on the contrary, create a low-pressure area. When the ship is close to a vertical obstacle, such as the bank, the pressure gradient will produce a couple of forces that will tend to stir the ship[6], as shown below. While the bank effect qualitatively explains the movements of the ship around $t = 7$ minutes in Figure 1, it is not sufficient to explain what happened previously.



This project will thus consist in simulating canal obstruction and some other variations on what happened. The goals are to understand why the Ever given got stuck, and what forces were present. We expect to observe and quantify the bank effect at around $t = 7$ minutes in Figure 1. We also whish to explain the reasons why the ship got this close to the shore in the first place. With the study of the typical forces, we also want to highlight possible feedback loops that happen when navigating in a restricted space with a large ship.

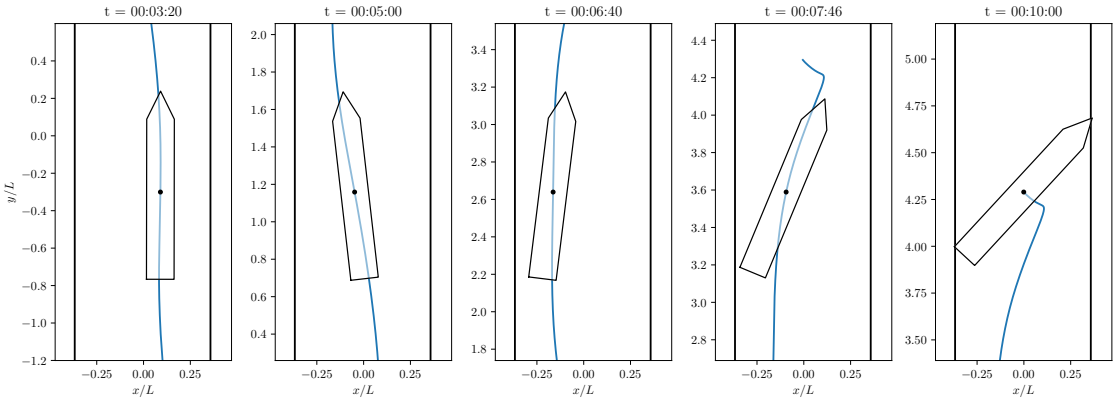


Figure 1: Snapshots of the ship positions during the incident

1.2 State of the art

This project mainly focuses on the study of the fluid forces on a solid, and the interactions that the solid and fluid have. Since the ship will be moving forward during most of the simulation, we will need to move the domain accordingly[1]. These two reasons naturally lead to the choice of using an ALE FEM model that moves along the ship vertically, and deforms to accommodate

[1] Or use a very large domain for the simulation but this is not an option that was retained.

the rotations and lateral movements. ALE models are commonly used to simulate problems where fluid-structure interaction is of key interest[7][8] due to their great interface capturing capabilities. Solids are described in a Lagrangian framework, while the Eulerian framework is most often used for fluids. The solid deformation is computed in a Lagrangian framework and then the mesh is moved using some kind of smoothing that aims at maintaining a reasonable mesh quality. The main challenge with ALE-FEM occurs when large deformations require remeshing to maintain an acceptable mesh quality, or changing the mesh topology. The solution to this problem involves re-meshing globally or locally, and projecting the solution on the new mesh[9]. The main drawback of a global remeshing is that the projection of the solution onto the new mesh introduces a lot of numerical diffusion. This also causes problems when we wish to keep a divergence-free field from one mesh to another, as it either adds constraints on the projection, or like in this report, produces non-physical behaviours[2] that may take some time to recover. Another drawback is that a global remeshing is difficult to apply in parallel. Local remeshing like done in [9], on the other had, requires a rather low-level interaction with the mesh generation algorithm, which was not easily feasible with dolfin.

2 Method

2.1 Simulation experiments

This project consists of three experiments. In the first experiment, we are interested in knowing the forces acting on the ship. The computation will be done by simulating the flow while enforcing the path of the boat through the canal. The forces and torque are then computed based on (1) and (2).

The second experiment will try to reproduce the path of the boat, but without enforcing the displacement a priori, simply by applying the forces computed previously on the boat. In this case, the boat movement is not known a priori, and the mesh displacement will be computed based on the forces, together with the fluid motion.

In the third experiment, we will try to tweak the forces and couples acting on the boat in order to prevent the ship from getting stuck. The goal is to see what kind of input would have been necessary to avoid the incident, and evaluate the real world feasibility of such inputs.

2.2 Mathematical model

The Suez canal incident only involved low speeds, therefore using the incompressible Navier-Stokes equations is a perfectly valid choice. The computation is based on a 2d model of the canal and the ship, with the ship being modelled as a hole in the domain Ω. The boundary conditions on both the ship and the shore are non-slip boundary conditions, with Neumann boundary conditions on the river. The resulting equations inside the domain are thus

$$\begin{cases} \dot{\boldsymbol{u}} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} + \nabla p - \Delta \boldsymbol{u} = 0 & x \in \Omega \\ \nabla \cdot \boldsymbol{u} = 0 & x \in \Omega \end{cases}$$

With boundary conditions

$$\begin{cases} \boldsymbol{u} = 0 & x \in \Gamma_{\text{shore}} \\ \boldsymbol{u} = \boldsymbol{u}_{\text{boat}} & x \in \Gamma_{\text{boat}} \\ pn - \nabla \boldsymbol{u} = 0 & x \in \Gamma_{\text{canal}} \end{cases}$$

Where Γ_{shore} is the portion of the boundary that is on the shore, Γ_{canal} is the artificial boundary that cuts the canal, and Γ_{boat} are the areas against the boat. The interaction between the ship and the water is governed Newton's second law.

[2] In this context those would be extremely strong pressure gradients.

2.3 Numerical model

For the numerical model, it is more convenient to work in an ALE framework, as it can easily capture both the mesh deformation and displacement. The numerical model will be similar to the one from lab 4, where we seek an approximation $(\boldsymbol{u}, p) \in V \times Q$ such that

$$(\dot{\boldsymbol{u}} + ((\boldsymbol{u} - \boldsymbol{w}) \cdot \nabla)\boldsymbol{u}, \boldsymbol{v})_\Omega - (p, \nabla \cdot \boldsymbol{v})_\Omega + (\nu \nabla \boldsymbol{u}, \nabla \boldsymbol{v})_\Omega + (\nabla \cdot \boldsymbol{u}, q)_\Omega + SD(\boldsymbol{u}, p; \boldsymbol{v}, q)_\Omega = 0$$

for all $(\boldsymbol{v}, q) \in V \times Q$, with $SD(\boldsymbol{u}, p; \boldsymbol{v}, q)_\Omega$ a stabilization term. All of the computations will use the non-dimensional form of the NSE. The scaling parameters are $U = 6\,\mathrm{m/s}$, $L = 400\,\mathrm{m}$, $\rho = 1000\,\mathrm{kg/m^3}$ and $\mu = 10^{-3}$ Pa · s, resulting in a Reynolds number $Re = 2.4 \times 10^9$. For the rest of the report, we will denote the dimensionless quantities with a "hat" : $\hat{\boldsymbol{u}} = \boldsymbol{u}/U$.

2.3.1 Computation of the forces

We also need to compute the forces and torque on the boat, which are given by

$$\boldsymbol{F}(\boldsymbol{u}, p) = \int_{\Gamma_{\text{boat}}} \nu \nabla \boldsymbol{u} \cdot n - pn = \underbrace{\mu U L_c Re}_{F_0} \cdot \int_{\hat{\Gamma}_{\text{boat}}} \hat{\nu} \nabla \hat{\boldsymbol{u}} \cdot n - \hat{p} n = F_0 \hat{\boldsymbol{F}}(\hat{\boldsymbol{u}}, \hat{p}) \tag{1}$$

$$\boldsymbol{\tau}(\boldsymbol{u}, p) = \underbrace{\mu U L L_c Re}_{\tau_0} \cdot \int_{\hat{\Gamma}_{\text{boat}}} \hat{\boldsymbol{r}}(\hat{x}) \times (\hat{\nu} \nabla \hat{\boldsymbol{u}} \cdot n - \hat{p} n) = \tau_0 \hat{\boldsymbol{\tau}}(\hat{\boldsymbol{u}}, \hat{p}) \tag{2}$$

where $\hat{\nu} = 1/Re$ and $\boldsymbol{r}(x) = x - x_p$ with x_p the point around which me compute the torque (the centre of mass of the boat). Note that we need to multiply the integrals by the draught $L_c = 14\,\mathrm{m}$ of the ship to get the correct force values.

2.4 Mesh deformation and remeshing

The meshes used in this report are generated using gmsh, as Dolfin's mshr had edge cases where the minimum mesh size was an order of magnitude smaller than desired, imposing a costly CFL bound.

The mesh deformation and displacement is handled by an elastic solid model where we prescribe the displacement at the boundaries. The displacement on the outside boundaries (i.e. the canal and the shore) is 0 on the x direction, and chosen such that the domain is centred around the ship's centre of mass in the y direction. The mesh deformation around the ship can be computed based on the displacement and rotation of the ship as a composition of a rotation and translation. The displacement and translation are either known a priori (experiment 1) or computed based on the fluid-solid interactions.

The large mesh deformations will inevitably lead to a degradation in the mesh quality, requiring a remeshing at some point. We implemented tree criteria to choose when to remesh:

1. The mesh quality goes below a certain threshold (0.2)
2. Some of the cells become too large
3. A change in the topology of the domain is required.

A change in the topology is required when the boat gets too close to the shore, or moves away from being too close to the shore. When the boat gets close to the shore, we either crop the boat to make sure that there is enough space for 2 cells between the boat and the boundary, or we extrapolate the boat to connect it to the shore. Having 2 cells between the ship and the shore is necessary to keep a divergence-free velocity. These cases are shown in Figure 2.

2.4.1 Projection problem

After remeshing, we project the solution $\hat{\boldsymbol{u}}$ onto the new mesh. This projection is done by Dolfin's project method which is orthogonal in the L_2 norm, but has the drawback that the projected solution is not divergence-free anymore. This in turn produces extremely strong unphysical oscillations

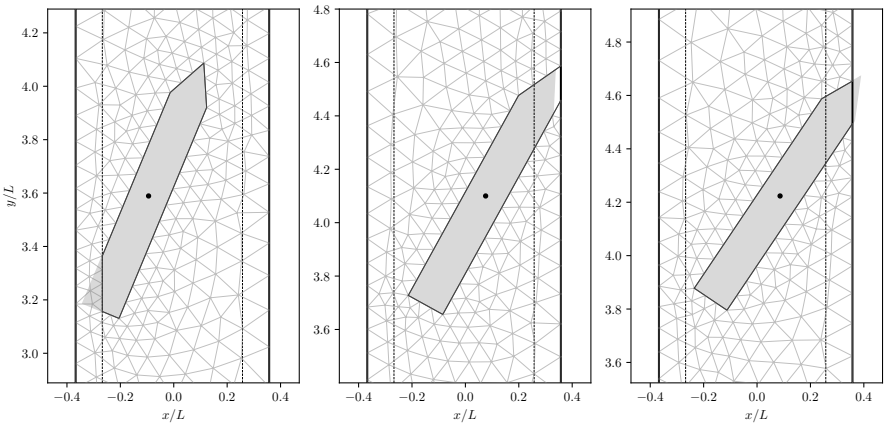


Figure 2: 3 different scenarios for dealing with the collision: the boat is cropped, the boat is extended, and the boat naturally makes contact with the boundary

in the pressure for a about 15 iterations, as shown on Figure 3. Fortunately, this number remains constant regardless of Δt, so the current code computes 15 iterations with $\Delta t = \Delta t_0/15$. During these 15 iterations, the computation of the force is ruined by the pressure gradient, so we estimate the forces by applying an aggressive smoothing on the force. Let $\tilde{F}$ be the force that we use and report, and $\hat{F}^{\text{comp}}$ the dimensionless computed force. then

$$\tilde{F}_{i+1} = (1 - w) \cdot \tilde{F}_i + w \cdot \hat{F}^{\text{comp}}$$

with w very small after a remesh.

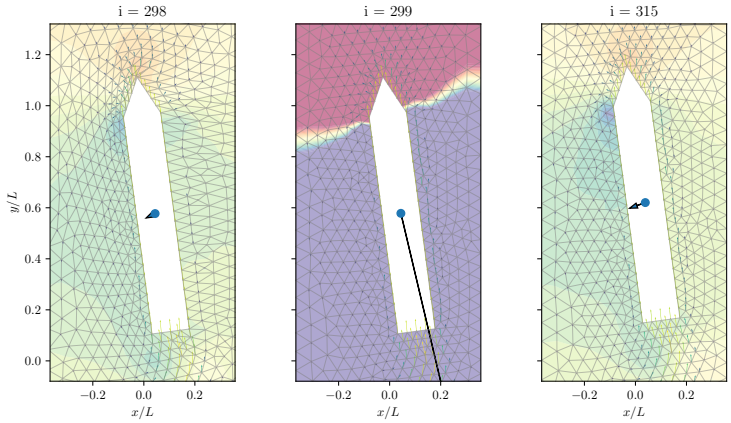


Figure 3: Pressure and velocity fields before and after a remesh. The arrow shows the computed fluid force on the ship.

2.5 Enforcing the forces

For experiments 2 and 3, we have to use the computed forces to move and accelerate the ship accordingly. The connection between the dimensionless force and acceleration can be expressed as

$$\hat{\boldsymbol{F}} = \frac{M_{\text{boat}}}{M_0} \cdot \frac{\mathrm{d}\hat{\boldsymbol{v}}}{\mathrm{d}\hat{t}}, \qquad M_0 = L^2 L_c \rho$$

$$\hat{\tau} = \frac{\hat{I} M_{\text{boat}}}{M_0} \cdot \frac{\mathrm{d}\hat{\omega}}{\mathrm{d}\hat{t}}, \qquad \hat{I} = \frac{1}{\hat{A}} \int_{\hat{\Omega}_{\text{boat}}} \mathrm{d}\hat{A}(\hat{x}^2 + \hat{y}^2)$$

The movement of the ship is then computed using Crank-Nicolson's method to integrate the acceleration and velocity. This is an implicit method which is solved iteratively together with the nonlinear iterations in the fluid computation. Doing that, however, leads to an unstable scheme, as the forces begin to oscillate and diverge. To solve this problem we use a smoothing similar to the one used after a remesh, leading to the following complete scheme:

$$\hat{v}_{i+1} = \hat{v}_i + \frac{\Delta t}{2}\left[(\tilde{F}_i + \tilde{F}_{i+1}) + (\hat{F}_i^{\text{other}} + \hat{F}_{i+1}^{\text{other}})\right]$$
$$\tilde{F}_{i+1} = (1-w)\cdot\tilde{F}_i + w\cdot\hat{F}_{i+1}^{\text{comp}}$$

The typical value of w is 0.5 in the second experiment and 0.2 in the third experiment. w is also massively scaled down after a remesh. The other (external) forces F^{other} are split into two components : the engine and the (truly) external forces. The engine force is assumed to be constant and in the direction the ship is heading to. Its magnitude is computed from the average longitudinal drag early in the simulation. The external forces are computed from the first experiment with

$$\frac{M_{\text{boat}}}{M_0}\cdot\frac{\mathrm{d}\hat{\boldsymbol{v}}}{\mathrm{d}\hat{t}} = \tilde{\boldsymbol{F}} + \hat{\boldsymbol{F}}^{\text{engine}} + \hat{\boldsymbol{F}}^{\text{extern}}$$

3 Results

3.1 First experiment : looking at the forces

The smoothed forces from experiment 1 are shown in Figure 4[3]. The solid lines indicate the

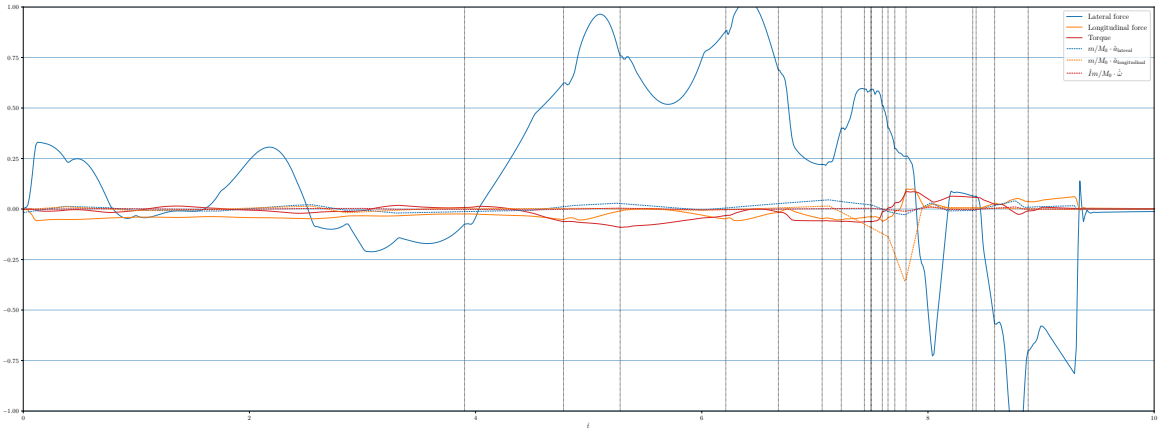


Figure 4: Comparison between the computed forces and torque and the expected forces and torque based on the acceleration.

computes forces, while the dashed lines are the expected forces based on the acceleration. The vertical lines indicate the moments where we do a remesh. The forces have been decomposed into their lateral and longitudinal component relative to the ship. This, among other things allows an easy computation of the mean drag, but il also clearly highlights that the side forces are *significantly* larger than expected. The longitudinal forces, on the other hand, roughly align with the acceleration in terms of magnitude if we correct for the drag/engine force. Looking at the expected and measured lateral forces in more detail, in Figure 5, we see that although the general direction of the force match the expectation, the magnitude of the measured force is more than an order of magnitude greater than expected. A similar story happens to the torque, although there the computed torque doesn't even have the expected shape.

[3]The plot is in a vector format so we highly recommend that the reader zooms into the pdf to have a better look.

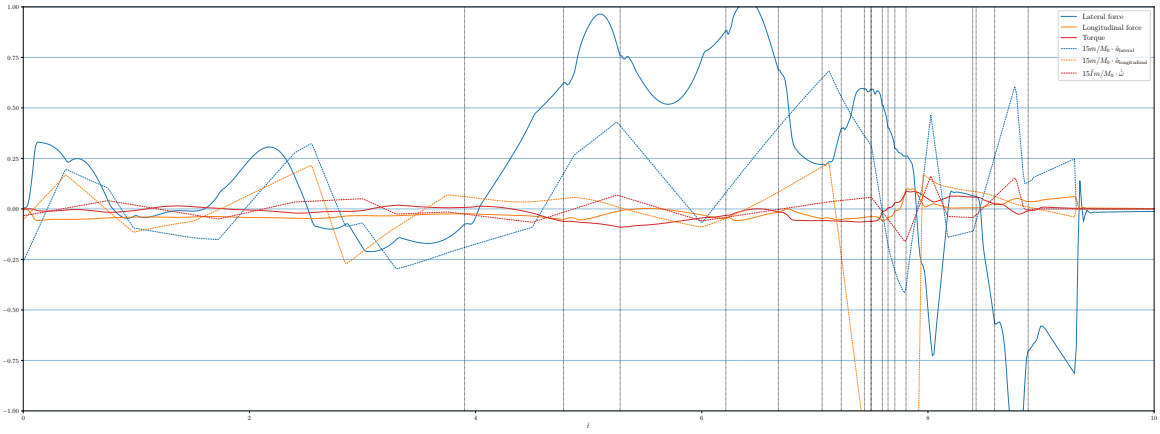


Figure 5: Comparison between the computed forces and torque and 15 times the expected forces and torque.

3.2 Second experiment : reproducing the events

While there clearly is a problem in the side forces from experiment 1, we can still use it and assume that there was an insanely high external side force that balanced the fluid forces. In this experiment we start the simulation at around $\hat{t} = 0.5$ to skip out the startup effetcs. Running the experiment results in a fairly similar path to the historical data. A series of snapshots comparing the simulation with the reality is shown on Figure 6.

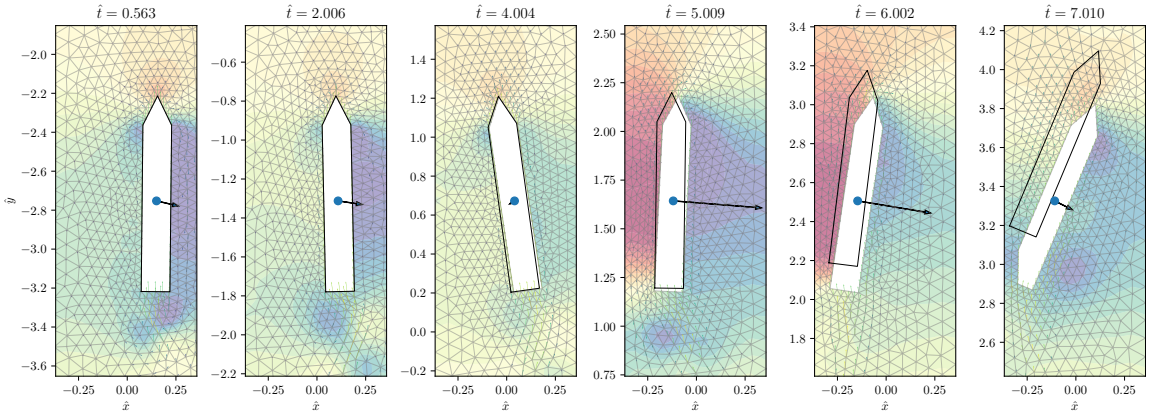


Figure 6: Comparison between the enforced path and the path simulated from the fluid forces

3.3 Third experiment : correcting the course

In light of the force problems, we decided to try to correct the course by only applying a small torque on the ship. When looking at the angular acceleration and inertia, we see that the maximum dimensionless torque has a magnitude of around 0.05, which is why we limited the applied torque to 0.02. Snapshots of the resulting path are shown in Figure 7. We see that we were able to avoid the collision with the shore until at least $\hat{t} = 8$. The simulation was not computed further in time as the external forces from experiment 1 start to be dominated by the contact forces, and it would be unrealistic to add them when there is no contact.

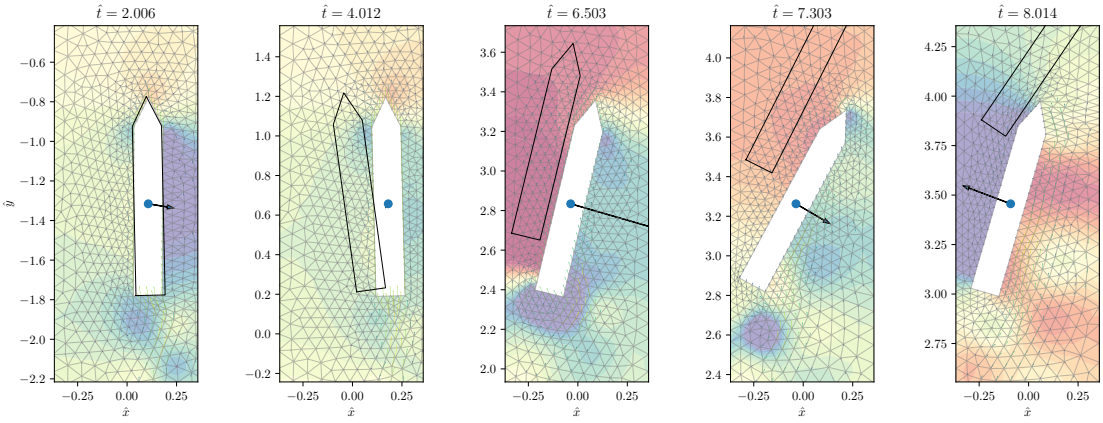


Figure 7: Comparison between the enforced path and the path simulated from the fluid forces with and added torque

4 Discussion

4.1 The problem of the lateral force

Before looking at the lateral force, it is worth noting that the longitudinal force is right about where we expect it to be. During the first part of the simulation, when the ship simply moves forward, the force is surprisingly constant and with a magnitude of $\hat{F} = 0.0375$. Converting this force to a dimensional form gives $F \approx 7500\,\text{kN}$ which, at $6\,\text{m/s}$ translates to $45\,\text{MW}$ of power. Given that the engine of the ship is capable producing $60\,\text{MW}$ for a speed of about $12\,\text{m/s}$, the $45\,\text{MW}$ instead of $\approx 15\,\text{MW}$ can be explained by the drag coefficient which is much smaller in reality than in this model. This also shows the correctness of the force computation and rules out an implementation error for the large side force.

While in theory the side force could be explained by external factors such as wind, the computed force would require wind speeds of around $450\,\text{km/h}$, which is clearly ruled out by the historical meteorological data. What mostly happens here is that our model is in 2D, which creates a significant difference compared to the reality: in reality, if the pressure is high on one side of the ship, the water can flow from one side to the other below the ship. Here, the only way to do so is for the water to go around the ship. A consequence is that if we force a small lateral displacement, the fluid in the model will respond with a very large pressure. In reality, however, such a displacement would correspond to a much lower pressure. The fact that the computed side forces and the expected side forces have the same shape also supports this conclusion. The ship likely experienced strong side winds that made it move, and this lateral movement translates to too large side forces in the model, but that are still in the right direction.

4.2 Consequences of the side force problem

A consequence of these strong lateral forces is that the bank effects have been largely overestimated. The strong side forces also explains why the computed torque is unusable. The computed torque mainly depends on the pressure on the side, and if those are massively increased, the errors in the torque will also be magnified.

Another surprising consequence is a strange restriction on the Δt used for experiments 2 and 3. In those, we observe that taking a significantly smaller Δt resulted in an unstable simulation, contrary to what is expected. This is due to the fact that the computed overestimated fluid forces and external forces have to balance each other out. If we use a smaller Δt, we will at some point get different values for the side forces, which creates an unbalance and a strong sideways acceleration. This in turn triggers a very high pressure increase resulting in an even more powerful side acceleration in the other direction, and the computation blows up.

4.3 Unexpected command to keep the ship on track

The third experiment showed that with minimal input, it was possible to correct the course of the ship and avoid the collision. The command that leads to the collision being avoided, however, is often very counter-intuitive: when the ship is close to one side of the canal, the "correct" way to guide it is to head to that same side of the canal! This is true for a significant part of the simulation, although not at the end, where when the ship heads to the right, we try to counter-steer it to the left.

One reason that may explain why we were able to correct the ship's course with a relatively small input is that when a ship starts turning, a positive feed back loops kicks in. Due to the inertia of the ship, a low pressure area is created inside the cure, which tends to turn the ship even more.

5 Conclusion

5.1 From the experiments

Unfortunately, the first experiment failed to quantitatively describe the forces and torque at play during the events. However, it still gives a good qualitative description of what happened if we keep in mind that the lateral forces are grossly overestimated. In combination with the second experiment, we can even conclude that the ship was subject to strong side winds, and know the direction of such winds, although not their magnitude.

Experiment 3 showed that is was indeed possible, and even easy to avoid this incident. We were able to correct it even in the context of insanely high lateral forces and torque! One can, however, make a very good counter-argument, stating that the commands were very unintuitive, but most importantly they were found using knowledge from the future. The sensitivity of the path relative to some small changes in the torque also supports this argument, stating that the ship is not always a stable system, and as such is difficult to control.

5.2 Future work and possible improvements

Regarding the last point, an interesting direction moving forward would be to try to find an algorithm that computes a command to keep the ship inside the canal based exclusively on past data.

The other thing that a future work should focus on is obviously the biggest problem of this report : the lateral forces. Here, the most obvious option would be to go for a 3d model. This is, however, a significantly harder problem to handle, as we would have to deal with the air water moving boundary, and it would also be very expensive in terms of processing power. An alternative to consider would be to model the ship as a porous media, and spend some time tweaking the porosity to get matching results from the expected and computed forces.

The second-biggest problem has to do with the remeshing. A very significant improvement could be done by changing the algorithm to a local mesh refinement/coarsening. This could massively improve the robustness of the code, and give a better insight on the forces just before the collision where the intense remeshing (and therefore smoothing) lowers the relevance of the results.

Finally, using a proper dataset for the historical data might also improve the quality of the results. These simulations use data obtained from the YouTube video in [10], followed by a cubic spline interpolation. The C_2 nature of the cubic spline ensures that the acceleration is continuous, but the transitions from one regime to the other still tend to disturb the computed forces.

References

[1] B. J. Harper, "Suez blockage is holding up $9.6bn of goods a day," *BBC News*, Mar 2021. [Online]. Available: https://www.bbc.com/news/business-56533250

[2] B. M.-A. Russon, "The cost of the Suez Canal blockage," *BBC News*, Mar 2021. [Online]. Available: https://www.bbc.com/news/business-56559073

[3] M. Safi, M. Farrer, and H. Smith, "Suez canal: Ever Given container ship freed after a week," *the Guardian*, Mar 2021. [Online]. Available: https://www.theguardian.com/world/2021/mar/29/suez-canal-attempt-re-float-ever-given-delay-salvage-tugboats

[4] "50 days after the Ever Given blocked the Suez canal, supply chains still have not recovered - CityAM," May 2021, [Online; accessed 6. Jun. 2021]. [Online]. Available: https://www.cityam.com/50-days-after-the-ever-given-blocked-the-suez-canal-supply-chains-still-have-not-recovered

[5] S. Writer, "Ever Given's Suez Canal blockage still disrupting global shipping," *Nikkei Asia*, May 2021. [Online]. Available: https://asia.nikkei.com/Business/Markets/Ever-Given-s-Suez-Canal-blockage-still-disrupting-global-shipping

[6] "Interaction – Knowledge Of Sea," Jun 2021, [Online; accessed 6. Jun. 2021]. [Online]. Available: https://knowledgeofsea.com/interaction

[7] Jeannette and J. Hoffman, "An interface-tracking unified continuum model for fluid-structure interaction with topology change and full-friction contact with application to aortic valves," *Int. J. Numer. Methods Eng.*, vol. n/a, no. n/a, Apr 2020.

[8] Y. Wang, A. Quaini, and S. Čanić, "A Higher-Order Discontinuous Galerkin/Arbitrary Lagrangian Eulerian Partitioned Approach to Solving Fluid–Structure Interaction Problems with Incompressible, Viscous Fluids and Elastic Structures," *J. Sci. Comput.*, vol. 76, no. 1, pp. 481–520, Jul 2018.

[9] G. Compère, J.-F. Remacle, J. Jansson, and J. Hoffman, "A mesh adaptation framework for dealing with large deforming meshes," *Int. J. Numer. Methods Eng.*, vol. 82, no. 7, pp. 843–867, May 2010.

[10] M.-P. com, "Perfect Simulation: Ever Given Accident in 2D plus 3D," May 2021, [Online; accessed 7. Jun. 2021]. [Online]. Available: https://www.marine-pilots.com/videos/234335-perfect-simulation-ever-given-accident-in-2d-plus-3d/?RL=Y

A Structure and usage of the code

The code is divided in 2 main blocks, one for getting and processing the data from the YouTube video[10] into a usable format, and then the fluid dynamics code which depends on the output of the first part. Running the fluid simulatiuon code as it is will create the figure of this report.

Running only the fluid dynamics part requires the following packages:

- `numpy`
- `matplotlib`
- `scipy`
- `dolfin` and `mshr`
- `numba`
- `gmsh`

The fluid sim code is in the file `main.py`, which heavily relies on `utils.py`. `utils.py` contains a set of custom functions that render the main file easier to read and understand. `utils.py` depends on the 3 following files:

- `raw data/smoothpath.csv`
- `raw data/smooth_hdg.csv`
- `raw data/smooth_coast.csv`

If one wishes to generate these 3 files, the following additional packages are required:

- `cv2`
- `pillow`
- `pytesseract`
- `xml`
- `parse`
- `codecs`

As well as the data file `raw data/map.xml` (which is an exported OSM map of the Suez Canal). The code should be run in the following order:

- `raw data/video2frames.py` : this will download the video (if it is not present in the directory) and analyse its frames to produce a list of coordinates of the ship. It will also run an OCR to get the additional data and save it to a file.
- `raw data/coastlineparse.py` : this will parse the XML map file and produce coordinates for the sides of the canal, and save them. It depends on the previous output for some visualization.
- `raw data/smoother.py` : this file will combine the long lists of coordinates of the ship and the canal to produce a cubic spline or Hermite approximation of the path and the coast. It will also rotate and translate the coordinates to have a canal with vertical sides in the straight section. These results are then saved in the 3 `csv` files used by the fluid sim.