Numerical modeling of a self propelled dolphin jump out of water

Michel Bergmann

Numerical modeling of a self propelled dolphin jump out of water

Michel Bergmann [1] [2]

[1] Inria, Memphis team, 200 avenue de la vielle tour, 33450 Talence, France
[2] Institut de Mathématiques de Bordeaux, 351, cours de la libération, 33405 Talence, France

E-mail: michel.bergmann@inria.fr

Abstract. A computational model is developed to investigate the jump of a self propelled dolphin out of water. This model relies on the Navier-Stokes equations where a fictitious domain approach with the volume penalization method is used for the fluid structure coupling, and the continuous surface force approach is used to model the water-air interface, the later being tracked in a level-set framework. The dolphin geometry is based on freely available data from the literature. While the body deformation is imposed, the leading linear and angular displacements are computed from the Newton's laws. Numerical simulations show that it is necessary to generate large propulsives forces to allow the jump out of water. When the dolphin is out of water, its trajectory follows a pure ballistic one.

1. Introduction

The study of underwater swimming animals is interesting to understand the possible effects related to the evolution, in particular to make link between the geometry of the body, the unsteady nature of the flow, and the unsteady swimming deformations imposed by muscles. The later can be optimized by using the energy enclosed in incoming vortices, in active (Gopalkrishnan et al. 1994, Liao et al. 2003, Fish & Lauder 2006, Triantafyllou et al. 2002) or passive (Beal et al. 2006) ways, or using vortices possibly generated by neighbor fishes (Bergmann & Iollo 2011, Maertens et al. 2017, Li et al. 2022, Gao & Triantafyllou 2018). Vortex induced control by the caudal tails of vortices generated along the fish body (Zhu et al. 2002) is also another mechanism to improve the swimming efficiency. The drag reduction and thrust generation are linked to the generation of an inverse Von Kármán street, as observed for simple manipulated circular cylinder wake flow (Bergmann et al. 2006).

A detailed comprehension of all these mechanisms may help to design new concept of Autonomous Underwater Vehicles (AUVs). The swimming of a fish has been considered, following the pioneer but controversial work of (Gray 1936), later on called the Gray's paradox, as being a very efficient mean of locomotion. A review from a fluid mechanics point of view of the Gray's paradox can be found in (Bale et al. 2014). Also, this type

of motion allows a maneuverability never reached by the previous generation of AUVs, of torpedo type (Bozkurttas et al. 2009).

As already mentioned, dolphin's kick swimming is an efficient way to move in water. Human swimmers, especially during competitions, try to mimic the dolphin's kick during the fully immersed stage (von Loebbecke, Mittal, Mark & Hahn 2009, von Loebbecke, Mittal, Fish & Mark 2009*b*). A comparison of human and cetaceans is given in (von Loebbecke, Mittal, Fish & Mark 2009*a*).

The swimming of an animal in water is a complex and beautiful phenomenon that is easy to observe in nature, but difficult to model numerically. The complexity comes from the reciprocal coupling between the fluid and the animal's body. This phenomenon is even more complex when the animal swims close to the surface, because it is necessary to take into account the deformation of the water surface and the forces related to the surface tension. The complexity is greatest when the animal crosses the surface, typically when it jumps out of the water. High-performance propulsion together with trajectory control to optimize the position at the early jump are required to jump out of water. The jumping dynamics of a variety of aquatic animals are described in (Chang et al. 2019), and of focus on the study of the archer fish jump can be found in (Mendelson & Techet 2020). Examples of jump spinning (twist) maneuvering can be found in (Fish et al. 2006).

The goal of this paper is thus to study the dolphin jump in a numerical framework. Only few numerical simulations of dolphin like swimming have been reported in the literature, most of them have been cited above. One reason is that it is difficult to define a realistic dolphin geometry. In this study we will use the freely available geometry of the *Lagenorhynchus obliquidens* dolphin proposed in (Tanaka et al. 2019). The other reason is, as previously mentioned, the complexity of the numerical modeling. Indeed, it is necessary to model the unsteady fluid structure interactions with large interface deformations and possible large density ratios (body and water interface deformations). In this study we consider a fictitious domain approach where the computational mesh, *i.e.* grid nodes, does not necessarily follow the interfaces. The dolphin body is considered with the Volume Penalization Method (Angot et al. 1999) and the fluid-structure interface is followed in a Lagrangian way, and the water-air interface is tracked in a Eulerian way using a Level set function (Osher & Sethian 1988, Sethian 1999). Even if it is not a dominant force at the dolphin's scale, we consider the surface tension using the Continuous Surface Force Method (CSF) originally introduced in (Brackbill et al. 1992). Other fictitious domain approaches can be used to compute large density ratios flows, an as for instance the one recently proposed in (Nangia, Griffith, Patankar & Bhalla 2019, Nangia, Patankar & Bhalla 2019). For simplicity reasons, no skin effects are considered in this study, even if they can play an important role in passive drag reduction (Pavlov 2006).

The paper is organized as follows. After having introduced the modeling and numerical methods in §2, including the description of the flow configuration, the governing equations and the associated numerical schemes, we will present the numerical

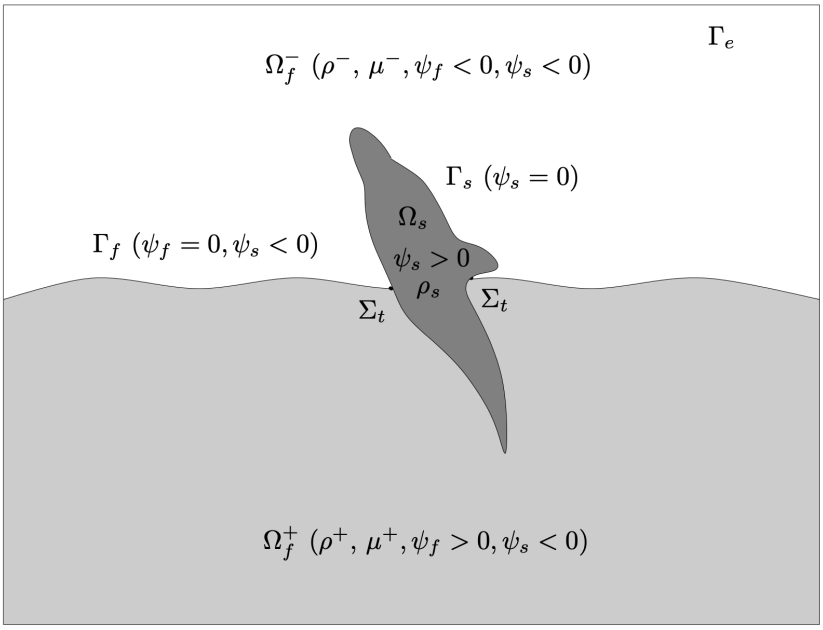


Figure 1. Two-dimensional sketch of the general three-dimensional flow configuration.

results for dolphin jumps out of water in §3. Some conclusions and perspectives are finally given in §4.

2. Modeling and numerical methods

2.1. Flow configuration

A sketch of the flow configuration is given in figure 1. The whole domain is $\Omega = \Omega_f^+ \cup \Omega_f^- \cup \Omega_s$, where Ω_s is the domain for the structure, *i.e.* the dolphin, Ω_f^+ is the domain filled with water, and Ω_f^- is the domain filled with air. The two fluid domains are separated by an interface Γ_f, and the structure and the fluid domains are separated by an interface Γ_s. The intersection of Γ_f and Γ_s is noted $\Sigma_t = \Gamma_f \cap \Gamma_s$. For three-dimensional problems, Σ_t could be a line, usually called the triple line. The external boundary of the domain is noted Γ_e. In what follows, the domain Ω is a three-dimensional Cartesian box, and the external boundary is $\Gamma_e = \Gamma_{top} \cup \Gamma_{bottom} \cup \Gamma_{left} \cup \Gamma_{right} \cup \Gamma_{front} \cup \Gamma_{back}$, where each boundary is defined as indicated by the subscripts. The density of the dolphin is ρ_s, the density and dynamic viscosity of water are ρ^+ and μ^+, and the density and dynamic viscosity of air are ρ^- and μ^-.

The different domains and interfaces introduced before also depend on time, and are mathematically defined by two level-set (Osher & Sethian 1988, Sethian 1999) scalar functions ψ_f and ψ_s. Here, the structure is arbitrarily defined by $\psi_s > 0$, the interface Γ_s by $\psi_s = 0$ and the fluids (water and air) by $\psi_s < 0$. Similarly, the water is arbitrarily defined by $\psi_f > 0$, the interface Γ_f by $\psi_f = 0$, and the air by $\psi_f < 0$.

2.2. Governing equations

Without loss of generality, the density $\rho \in \mathbb{R}$ and dynamic viscosity $\mu \in \mathbb{R}$ of the fluids can be written:

$$\rho \equiv \rho(\psi_f) = \rho^- + H(\psi_f)(\rho^+ - \rho^-), \tag{1}$$

$$\mu \equiv \mu(\psi_f) = \mu^- + H(\psi_f)(\mu^+ - \mu^-). \tag{2}$$

where H is the Heaviside function, *i.e.* $H(x < 0) = 0$, $H(x > 0) = 1$, and we can define $H(x = 0) = 0.5$.

The velocity fields is $\boldsymbol{u} =\in \mathbb{R}^3$, $p \in \mathbb{R}$ is the pressure field and $\boldsymbol{g} \in \mathbb{R}^3$ is the gravity acceleration vector. The viscous part of the stress tensor is $D(\boldsymbol{u}) = \dfrac{\boldsymbol{\nabla u} + \boldsymbol{\nabla}^T \boldsymbol{u}}{2}$. In what follows, we note $\boldsymbol{u} = (u, v, w)$, where u, v and w denote the velocity components in the x, y and z directions respectively. The incompressible Navier-Stokes equations for both fluids in domain $\Omega_f^+ \cup \Omega_f^-$ are:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \boldsymbol{\nabla})\boldsymbol{u} = -\frac{1}{\rho}\boldsymbol{\nabla} p + \frac{1}{\rho}\boldsymbol{\nabla} \cdot 2\mu D(\boldsymbol{u}) + \boldsymbol{g} \text{ in } \Omega_f^+ \cup \Omega_f^-, \tag{3}$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0 \text{ in } \Omega_f^+ \cup \Omega_f^- -, \tag{4}$$

with initial conditions $\boldsymbol{u}(\boldsymbol{x}, t = 0) = \boldsymbol{u}_0$ and $p(\boldsymbol{x}, t = 0) = p_0$, boundary conditions on the external boundary for velocity $\boldsymbol{u(x, t)}$ and pressure $p(\boldsymbol{x}, t)$ for $\boldsymbol{x} \in \Gamma_e$, boundary conditions on the structure boundary Γ_s, and conditions through the bi-fluid interface Γ_f.

On the fluid-structure interface Γ_s, we have:

$$\boldsymbol{u}(\boldsymbol{x}, t) = \widehat{\boldsymbol{u}}(\boldsymbol{x}, t) \text{ on } \Gamma_s. \tag{5}$$

where the velocity $\widehat{\boldsymbol{u}}(\boldsymbol{x}, t)$ will be described later on.

Two jump conditions have also to be satisfied throught the bi-fluid interface Γ_f. The first kinematic condition is:

$$[\boldsymbol{u}] = 0 \text{ throught } \Gamma_f. \tag{6}$$

The second dynamic jump condition traduces the equilibrium between pressure forces, the friction forces and the surface tension:

$$[-pI + 2\mu D(\boldsymbol{u})] \cdot \boldsymbol{n} = \sigma\kappa\boldsymbol{n} \text{ throught } \Gamma_f, \tag{7}$$

where σ est the value of the surface tension, κ is the curvature of the interface Γ_f and $\boldsymbol{n}$ is the unit normal to interface Γ_f pointed to the air.

2.3. Fictitious domain approach

The numerical resolution of the Navier-Stokes equations (3) and (4) requires a discretization on a mesh. The difficulty is that the interfaces Γ_f and Γ_s, and thus the domains Ω_f^+, Ω_f^- and Ω_s, are time dependent. Different approaches to taken into account interface conditions (5), (6) and (7) can be envisioned.

This first class of methods are based on body-fitted grids. In these methods, degrees of freedom are put on the interfaces, and it is thus possible to impose directly the interface conditions (5), (6) and (7). Mesh deformation can be handle with arbitrary Lagrangian-Eulerian (ALE) method. These methods are accurate, but require mesh adaptation and a mesh partitioning for parallel computations.

A second class of methods, adopted in this study, is based on fictitious domain approaches. In these approaches, the interfaces and associated domains, do not covered the same mesh nodes at each time step: the interface can cross a fixed mesh, and no interface markers are thus required. Simple meshes like Cartesian ones can be used. The drawback is that the accuracy at interfaces can be degraded, and an extra work has to be performed to model the interfaces.

The goal of fictitious approaches is to add extra force-like terms in the momentum equations (3) to take into account the conditions (5), (6) and (7). The condition on the fluid-structure interface is modeled with an extra term $\boldsymbol{s}$ and the two conditions on the bi-fluid interface are modeled with the extra term $\boldsymbol{f}$.

The system (3), (4), (5), (6) and (7) is thus recasted in a system written in the whole domain Ω:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\boldsymbol{\nabla})\boldsymbol{u} = -\frac{1}{\rho}\boldsymbol{\nabla} p + \frac{1}{\rho}\boldsymbol{\nabla}\cdot 2\mu D(\boldsymbol{u}) + \boldsymbol{g} + \boldsymbol{s} + \boldsymbol{f} \quad \text{in } \Omega, \tag{8}$$

$$\boldsymbol{\nabla}\cdot\boldsymbol{u} = 0 \quad \text{in } \Omega, \tag{9}$$

The methods used to model the terms $\boldsymbol{s}$ and $\boldsymbol{f}$ have been inspired by the Immersed Boundary Method (IBM) introduced by Peskin (Peskin 1972). In this continuous framework, an infinite force applied on a Dirac support function is regularized around the interface. This is mainly done for numerical reasons that will be explained later on.

In this study, the fluid structure interface is modeled using the Volume Penalization introduced in (Angot et al. 1999) §2.3.1, and the bi-fluide interface with surface tension is modeled using the Continuous Surface Force (CSF) method introduced in (Brackbill et al. 1992) §2.3.2.

2.3.1. Modeling of the conditions on the fluid-structure interface Γ_s Among the most popular fictitious approches are the Immersed Boundary Methods (IBM) originally introduced in (Peskin 1972) and later on used in several studies (Mittal & Iaccarino 2005, Mittal et al. 2008, Shirgaonkar et al. 2009) and the Volume Penalization (VP) Method introduced in (Angot et al. 1999) and used for instance in biolocomotion problems (Bergmann & Iollo 2011, Bergmann, Iollo & Mittal 2014, Bergmann & Iollo 2016). Another approche combining IBM and VP has been developed in (Bergmann, Hovnanian & Iollo 2014). Here, we used the VP method with

$$\boldsymbol{s} = \frac{\chi}{K}(\widehat{\boldsymbol{u}} - \boldsymbol{u}), \tag{10}$$

where $K \ll 1$ is the penalization parameter, $\chi = H(\psi_s)$ is the characteristic function, and $\widehat{\boldsymbol{u}}$ is the velocity of the body (on the boundary and inside the body). In domain $\Omega^{\pm}$ we have $\psi_s < 0$ and thus $\chi = 0$, and the Navier-Stokes equations are recovered. In

domain Ω_s, we have $\psi_s > 0$, and thus $\chi = 1$, and the dominant term in (3) is $\frac{\chi}{K}(\widehat{\boldsymbol{u}} - \boldsymbol{u})$ leading to $\boldsymbol{u} \to_{K\to 0} \widehat{\boldsymbol{u}}$ in Ω_s. Here, we chose $K = 10^{-8}$. The computation of $\widehat{\boldsymbol{u}}$ will be detailed in §2.5.

2.3.2. Modeling of the conditions on the bi-fluid interface Γ_f The CSF (Continuum Surface Force) has been originally introduced in (Brackbill et al. 1992) and developed in (Sussman et al. 1994). The CSF is to model the surface tension term as an extra volume force $\boldsymbol{f}$ in the momentum equation. The jump

$$[-pI + 2\mu D(\boldsymbol{u})] \cdot \boldsymbol{n} = \sigma\kappa\boldsymbol{n} \text{ throught } \Gamma_f, \tag{11}$$

is considered in an integral form and we have:

$$\boldsymbol{f} = \frac{1}{\rho}\sigma\kappa\delta\boldsymbol{n}. \tag{12}$$

where σ is the surface tension, κ the curvature of the interface, $\boldsymbol{n}$ the normal to the interface pointing to the air, and δ the Dirac distribution.

The normal $\boldsymbol{n}$ and the curvature κ are:

$$\boldsymbol{n} = \frac{\boldsymbol{\nabla}\psi_f}{\|\boldsymbol{\nabla}\psi_f\|}, \tag{13}$$

$$\kappa = \boldsymbol{\nabla} \cdot \left(\frac{\boldsymbol{\nabla}\psi_f}{\|\boldsymbol{\nabla}\psi_f\|}\right). \tag{14}$$

The problem is relaxed by regularization of the Dirac distribution δ in a narrow band with width ϵ

$$H^\epsilon(\psi_f) = \begin{cases} 0 & \text{si } \psi_f < -\epsilon, \\ \frac{1}{2}\left(1 + \frac{\psi_f}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi\psi_f}{\epsilon})\right) & \text{si } |\psi_f| \le \epsilon, \\ 1 & \text{si } \psi_f > \epsilon. \end{cases} \tag{15}$$

The regularized Dirac distribution is:

$$\delta^\epsilon(\psi_f) = \frac{\mathrm{d}H^\epsilon(\psi_f)}{\mathrm{d}\psi_f} = \begin{cases} 0 & \text{si } |\psi_f| > \epsilon, \\ \frac{1}{2\epsilon}\left(1 + \cos(\frac{\pi\psi_f}{\epsilon})\right) & \text{si } |\psi_f| \le \epsilon. \end{cases} \tag{16}$$

The extra term due to the surface tension is thus finally

$$\boldsymbol{f} = \frac{1}{\rho}\sigma\kappa\delta^\epsilon\boldsymbol{n}. \tag{17}$$

Finally, due to the continuous framework introduced beforehand, the jump $[\boldsymbol{u}] = 0$ throught Γ_f in equation (6) is naturally satisfied.

2.4. Interface tracking

The model will be closed with the computations of functions ψ_s and ψ_f. The definition of these functions is not unique, and due to its hyper-regularity we chose ψ_f to be

the signed distance function, *i.e.* $\|\boldsymbol{\nabla}\psi_f(\boldsymbol{x})\| = 1 \quad \forall \boldsymbol{x}$. Let $d(\boldsymbol{x}) \geq 0$ be the minimal distance from a given point $\boldsymbol{x}$ to the interface Γ_f, we have:

$$\begin{cases} \psi_f(\boldsymbol{x}) = 0 \text{ if } \boldsymbol{x} \in \Gamma_f, \\ \psi_f(\boldsymbol{x}) = +d \text{ if } \boldsymbol{x} \in \Omega_f^+, \\ \psi_f(\boldsymbol{x}) = -d \text{ if } \boldsymbol{x} \in \Omega_f^-. \end{cases} \tag{18}$$

The same holds for ψ_s. The signed distance allows to simplify the computation of the normal and the curvature of the interface, and determine in a more precise way the distance ε defining the narrow band for CSF regularization.

These interfaces, represented by functions ψ_f and ψ_s, can be transported in Eulerian or Lagrangian ways. Due to possible large deformations and break downs, the bi-fluid interface Γ_f will be defined with a level set function that is able to track such interface with topology changes. The level set ψ_f is thus transported by:

$$\frac{\partial \psi_f}{\partial t} + \boldsymbol{w} \cdot \boldsymbol{\nabla}\psi_f = 0 \text{ in } \Omega_f^{\pm}, \tag{19}$$

where $\boldsymbol{w}$ can be any desired velocity fields satisfying $\boldsymbol{w}(\boldsymbol{x}, t) = \boldsymbol{u}(\boldsymbol{x}, t) \ \forall \boldsymbol{x} \in \Gamma_f$. For simplicity reasons we chose $\boldsymbol{w}(\boldsymbol{x}, t) = \boldsymbol{u}(\boldsymbol{x}, t) \ \forall \boldsymbol{x} \in \Omega$, but other choices like the extension velocities can be used (Adalsteinsson & Sethian 1999).

The signed distance behavior of the function ψ_f can be lost integrating (19), and it is thus necessary time to time to reinitialize ψ_f to recover the signed distance function $\|\boldsymbol{\nabla}\psi_f(\boldsymbol{x})\| = 1$. One of the most popular algorithm is the first order fast marching (Sethian 1996, Sethian 1999), extended later to higher orders (Ahmed et al. 2011).

Another class of method was introduced by Sussman (Sussman et al. 1994). The goal is to solve a partial differential equation to recover the property $\|\boldsymbol{\nabla}\psi_f(\boldsymbol{x})\| = 1$. This relaxation method is:

$$\frac{\partial \psi_f}{\partial \tau} + sign(\psi_f^0)(|\nabla \psi_f| - 1) = 0, \tag{20}$$

$$\psi_f(\boldsymbol{x}, \tau = 0) = \psi_f^0 = \psi_f(\boldsymbol{x}, t), \tag{21}$$

for a temporal horizon $0 \leq \tau \leq \tau_F$ such that $|\nabla \psi_f(\boldsymbol{x}, \tau_F)| = 1$. The sign function is

$$sign(\psi_f) = \begin{cases} 1, & \text{if } \psi_f > 0, \\ 0, & \text{if } \psi_f = 0, \\ -1, & \text{if } \psi_f < 0, \end{cases} \tag{22}$$

or a regularized version of it.

In what follows, the dolphin surface Γ_s will be approximated by a mesh (*i.e.* with markers), and it is thus more convenient to use a Lagrangian transport for any mesh point $\boldsymbol{x}_h$:

$$\frac{\mathrm{d}\boldsymbol{x}_h}{\mathrm{d}t} = \widehat{\boldsymbol{u}}_h, \tag{23}$$

where the definition and computation of $\widehat{\boldsymbol{u}}$ will be defined in §2.5.

The signed distance function is recovered computing the minimal distance to the interface:

$$\psi_s(\boldsymbol{x}) = \min_{\boldsymbol{y}\in\Gamma_s} \|\boldsymbol{x}-\boldsymbol{y}\|_2 S(\boldsymbol{x}). \tag{24}$$

where $S(\boldsymbol{x})$ denote the sign applied on a point $\boldsymbol{x}$, with $S(\boldsymbol{x}) > 0$ inside the body, and $S(\boldsymbol{x}) < 0$ outside. This sign function can be computed with simple geometric arguments, from the outward normal to the body.

The triple line Σ_t, defined as being the intersection between Γ_s and Γ_f, has to be modeled. Indeed, without special treatment, due to non-slip boundary condition on Γ_s, the line would never move on Γ_s. In this study we choose the macroscopic model introduced by Cox (Cox 1986). A local force proportional to the deviation of the equilibrium angle (Patankar 2003) is added to allow a displacement of the triple line.

Finally, since the flows under consideration are turbulent, a turbulence model introduced in (Vreman 2004) is added.

2.5. Dolphin model

The geometry of the Dolphin is built from the *stl* file given in the supplementary material of (Tanaka et al. 2019). Indeed, the authors have performed a very productive work given lot of material to work with. The geometry is reproduced in Figure 2. The two first figures are the exact geometry given by (Tanaka et al. 2019), and the two last figures are the geometry a little bit remodeled to fit our numerical framework with regular surface mesh.

The body velocity is

$$\widehat{\boldsymbol{u}}(\boldsymbol{x},\, t) = \overline{\boldsymbol{u}}(\boldsymbol{x},\, t) + \boldsymbol{u}^{\theta}(\boldsymbol{x},\, t) + \widetilde{\boldsymbol{u}}(\boldsymbol{x},\, t) \quad \forall \boldsymbol{x} \in \Omega_s, \tag{25}$$

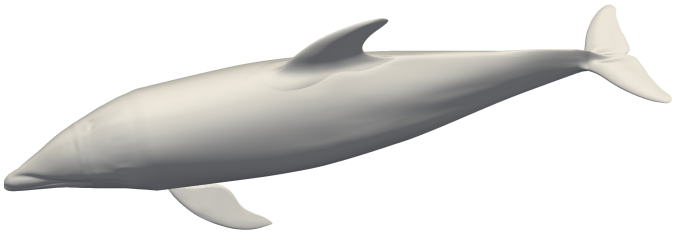
where $\overline{\boldsymbol{u}}(\boldsymbol{x},\, t)$ is the linear velocity, $\boldsymbol{u}^{\theta}(\boldsymbol{x},\, t)$ is the angular velocity and $\widetilde{\boldsymbol{u}}(\boldsymbol{x},\, t)$ is the deformation velocity. While the deformation $\widetilde{\boldsymbol{u}}(\boldsymbol{x},\, t)$ has to be imposed by swimmer muscles, the linear and angular velocities are the results of the loads generated by the fluid on the body, and are computed from the Newton's laws.

In order to avoid adding extra forces and torques due to deformation, a Procrustes analysis is performed. The deformation must not introduce any linear and angular displacements. We thus compute the linear and angular displacements induced by the imposed deformation, and subtract them to the final admissible deformations. This is actually done in Figure 4.

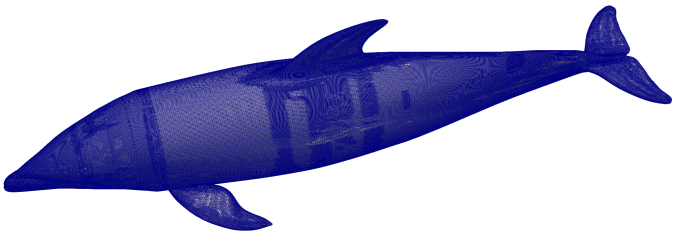
The deformation velocity $\widetilde{\boldsymbol{u}}(\boldsymbol{x},\, t)$ can be easily computed following surface markers in a Lagrangian way, after having performed the procrustes analysis. To recover $\widetilde{\boldsymbol{u}}(\boldsymbol{x},\, t)$ values in Ω_s, interpolation is performed from values at the boundary Γ_s.

The swimming law is defined by a deformation of the swimmer midline, also called the backbone. As we will see later on, the dolphin jump can be decomposed in three steps.

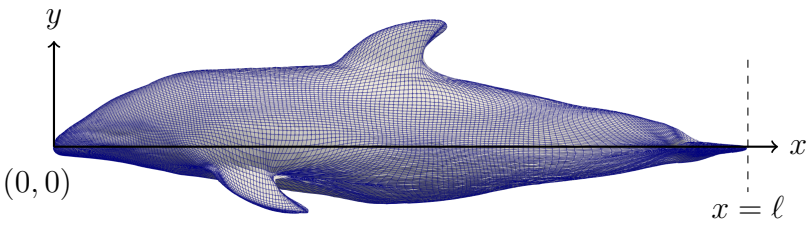
- The first one is an underwater generation of a propulsive force using a periodic swimming with a large amplitude and frequency.



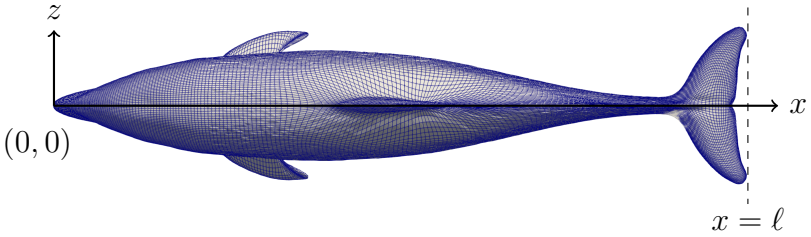
(a) Original shape.



(b) Original mesh.



(c) Side view of the modified mesh used in this study.



(d) Top view of the modified mesh used in this study.

Figure 2. Geometry of the Dolphin built with the *stl* file from supplementary material of Tanaka et al. (2019). The two bottom figures show the mesh used in this study.

- The second one corresponds to the jump out of the water, *i.e.* when the dolphin is in the air: the dolphin just imposes (active or passive) a constant curvature along the midline.
- The third and last step is another periodic swimming law with a small amplitude when the dolphin is back in water.

We consider a backbone deformation in the plan $(0, x, y)$ of the third figure in figure 2, where point O is at the front head of the dolphin and x positive to the right. We consider moreover that the midline for the steady body is $0 \leq x \leq \ell$, $y = 0$, where ℓ is the length of the swimmer.

Most of the fishes impose the following periodic swimming law (Barrett et al. 1999):

$$y(x,\, t) = a(x)\sin(k\,x - \omega\,t), \tag{26}$$

with a constant phase velocity $c_p = \omega/k$, where $k = 2\pi/\lambda$ is the wavenumber, corresponding to wavelength λ, ω is the circular frequency of oscillations, and where $a(x)$ is the envelop defined by $a(x) = A/2(c_0 + c_1(x-1) + c_2(x^2-1))$. Other kinds of swimming laws can be found in (Smits 2019) and (Han et al. 2020). The maximal tail amplitude $A/2$ is an important parameter for the swimming (Lighthill 1970). Here, we chose $c_0 = 1$, $c_1 = -0.825$ and $c_2 = 1.625$ for a unit length fish with $0 \leq x \leq 1$. These parameters are used to mimic a thunniform like swimming. Several midline deformations are presented in Figure 3 for one stroke, with $A = 0.33$.

Instead imposing directly the swimming law (26), it is more simple to impose a curvature $\kappa(s)$ along the midline with curvilinear abscissa s. The curvature corresponding to the swimming law $y(s,\, t) = a(s)\sin(k\,s - \omega\,t)$ for the first and third stage of the jump is:

$$\kappa(s,t) = \frac{\frac{\partial^2 y}{\partial s^2}}{\left(1 + \left(\frac{\partial y}{\partial s}\right)^2\right)^{\frac{3}{2}}}. \tag{27}$$

The general curvature imposed on the midline is:

$$\kappa(s,\, t) = p(t)\kappa(s,\, t) + q(t)\overline{\kappa}, \tag{28}$$

where $\overline{\kappa}$ is the curvature imposed for the jump, *i.e.* when the dolphin is in the air, and $0 \leq p(t) \leq 1$ and $0 \leq p(t) \leq 1$ are two functions, possibly piecewise defined, allowing to go smoothly from step 1 to step 2 and from step 2 to step 3.

This swimming law is apply to the dolphin midline (the backbone). All body surface mesh nodes are displaced with respect to Euler Bernoulli beam deformation, *i.e.* each orthogonal section to the backbone for the undeformed shape remains orthogonal to the backbone during the deformation. Eight snapshots of the deformed body shape corresponding to (28) over one periodic stroke ($p = 1$ and $q = 0$) are presented in Figure 4. Note that a procrustes analysis is performed.

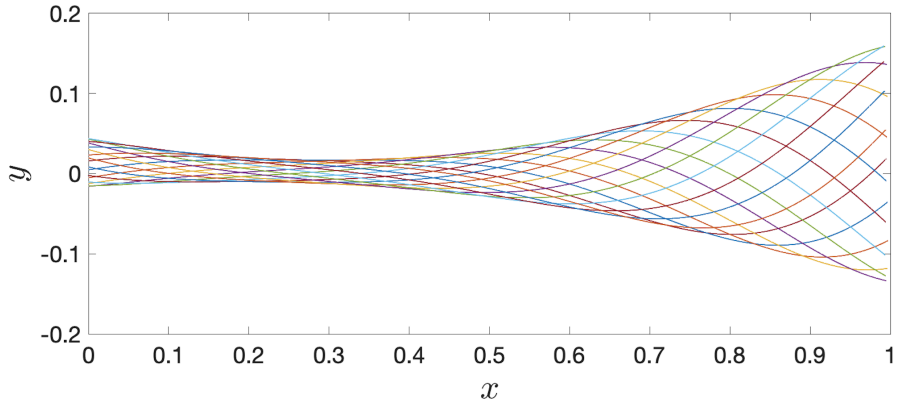
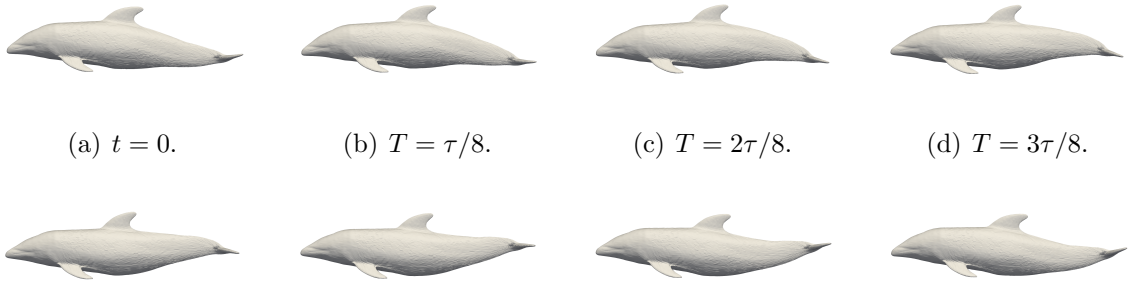


Figure 3. Periodic swimming law: backbone deformation with $A = 0.33$.



(a) $t = 0$. (b) $T = \tau/8$. (c) $T = 2\tau/8$. (d) $T = 3\tau/8$.

(e) $T = 4\tau/8$. (f) $T = 5\tau/8$. (g) $T = 6\tau/8$. (h) $T = 7\tau/8$.

Figure 4. Temporal evolution of the dolphin shape over one stroke period τ. Should be read from left to right, top to bottom. $t = 0$, $T = \tau/8$, $T = 2\tau/8$, $T = 3\tau/8$, $T = 4\tau/8$, $T = 5\tau/8$, $T = 6\tau/8$, $T = 7\tau/8$

Finally, the linear and angular motions are obtained from the Newton's law

$$\begin{cases} m\dfrac{\mathrm{d}\overline{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{F}_{ext}, \\ \dfrac{\mathrm{d}J\boldsymbol{\omega}}{\mathrm{d}t} = \boldsymbol{M}_{ext}, \end{cases} \tag{29}$$

where m and J are the mass and inertia matrix of the dolphin, and $\overline{\boldsymbol{u}}$ and $\boldsymbol{\omega}$ denote the linear and angular velocities.

The forces and the torques are computed by

$$\boldsymbol{F}_{ext} = -\int_{\Gamma_s} \mathbb{T}(\boldsymbol{u},\, p)\, \boldsymbol{n}\, \mathrm{d}\boldsymbol{x} + \boldsymbol{g}, \tag{30}$$

$$\boldsymbol{M}_{ext} = -\int_{\Gamma_s} \boldsymbol{r} \wedge \mathbb{T}(\boldsymbol{u},\, p)\, \boldsymbol{n}\, \mathrm{d}\boldsymbol{x}, \tag{31}$$

where $\mathbb{T}(\boldsymbol{u}, p) = -p\boldsymbol{I} + \mu(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)$ is the stress tensor, $\boldsymbol{n}$ is the unit outward vector to Γ_s, and $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}_G$ with $\boldsymbol{x}_G$ the dolphin center of mass.

The rotation velocity is given by $\boldsymbol{u}^\theta = \boldsymbol{\omega} \wedge \boldsymbol{r}$.

Note that other methods can be used to computed the forces and torques in fictitious domain approach when no surface markers are available. For instance, the surface integrals in (30) and (31) can be transformed in a control volume integrals (Nangia et al. 2017).

2.6. Numerical resolution

The Navier-Stokes equations (8) and (9), and associated equations like transport and reinitialization of the level set function, are discretized in space on an uniform Cartesian mesh using finite differences method. The temporal discretization of the Navier-Stokes equations is based on a fractional step method introduced in (Chorin 1968) and (Temam 1969). The numerical method used in this paper follows (Bergmann & Iollo 2011, Bergmann, Hovnanian & Iollo 2014, Bergmann, Iollo & Mittal 2014, Bergmann & Iollo 2016) and we recall thus only the main steps.

All the differential operators are discretized in space using centered second order schemes, except the convective terms where up-winding is necessary for stability reasons. The convective terms in the momentum equations are discretized with a classical third order finite differences scheme, while the convective term in the level set transport equation (19) is discretized with a WENO5 scheme (Liu et al. 1994, Jiang & Shu 1996). A total variation diminishing third order Runge-Kutta (RK3-TVD) scheme is used to discretized equation (19) in time. The reinitialization equation (20) is solved using the the subcell fix method proposed in (Russo & Smereka 2000) with extensions developed in (Luddens et al. 2015). It is not necessary to perform a reinitialization step at each time step, and we use the criterion introduced in (Luddens et al. 2015) to decide when it is necessary to perform a reinitialization of the level set function.

The integrals over Γ_s involved to evaluate the forces and torques are discretized using the body the surface mesh (that is a tessellation of the surface Γ_s). Since the dolphin's surface mesh nodes do not coincide with the fluid mesh, the stress tensor is interpolated from fluid mesh to the body surface mesh.

2.7. Numerical validations

The numerical approach presented above has already been validated onto several benchmark problems in our previous publications. Several validations in three dimensions complement numerous validations already performed for two dimensional problems. Our previous validations consider mono-fluid problems, *i.e.* without bi-fluid interface. A first validation is the sedimentation of a sphere (Bergmann, Iollo & Mittal 2014) for several sphere diameters and fluid viscosities. For bio-inspiration problems involving large deformations, another validation has been performed in (Bergmann & Iollo 2016) onto an eel swimming originally proposed by (Kern & Koumoutsakos 2006)

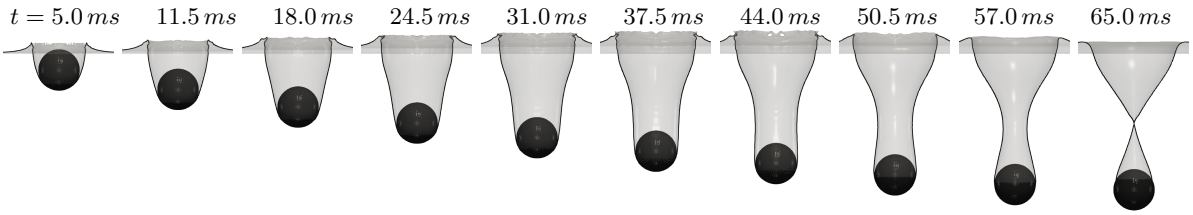


Figure 5. Image sequences showing the water-entry cavity formed by the sphere. Times since the sphere center passed the free surface ($t = 0$) are shown until the Pinch-Off at $t = 65\,ms$.

and studied numerically thereafter in (Bhalla et al. 2013) and (Bhalla et al. 2014). For both validation problems, the numerical results are in good agreements with reference ones.

In what follow, a three-dimensional bi-fluid validation is performed for the entry of a sphere in water. This problem has been studied experimentally (Aristoff et al. 2010) and (Truscott et al. 2014), and numerically with the commercial software ABAQUS in (Ahmadzadeh et al. 2014).

A sphere with diameter $D = 0.0254\,m$ impacts the water surface at $V_0 = 2.17\,m/s$. The density of the sphere is $\rho_s = 1.14\rho^+$. The densities and dynamic viscosities of the fluids are $\rho^+ = 1000\,kg/m^3$ and $\mu^+ = 10^{-3}\,Pa.s$ for the water, and $\rho^- = 1.2\,kg/m^3$ and $1.87 \cdot 10^{-5}\,Pa.s$ for the air. The surface tension is $\sigma = 0.0728\,N/m$.

The computational domain under consideration is a Cartesian box with $x \in [-6, 6]\,cm$, $y \in [-20, 10]\,cm$ and $z \in [-6, 6]\,cm$. At rest, the air-water interface Γ_f is located at $y_I = 0\,cm$, and the initial position of the center of mass of the sphere is $\boldsymbol{x}_G = (x_G,\, y_G,\, z_G) = (0,\, D/2,\, 0)\,m$.

The computational domain is discretized by $240 \times 600 \times 240 \approx 35 \cdot 10^6$ cells, and the step size is $h = \Delta x = \Delta y = \Delta z = 0.05\,cm$. The time step is defined with a CFL condition equal to 0.3.

Non-slip boundary conditions are applied on all external boundaries. The flow is initially at rest, *i.e.* $\boldsymbol{u}(0, \boldsymbol{x}) = \boldsymbol{0}$.

Figure 5 presents the image sequences showing the water-entry cavity formed by the sphere until $t = 65\,ms$ where the Pinch-Off, that is characterized by a cavity closure event, occurs. Results are in good agreement with experimental results (Aristoff et al. 2010) where the Pinch-Off is observed at $t = 63.5\,ms$, and with the numerical results (Ahmadzadeh et al. 2014) where the Pinch-Off is observed at $t = 64.9\,ms$.

A time history comparison of experimental (Aristoff et al. 2010), numerical ABAQUS simulation (Ahmadzadeh et al. 2014) and present numerical results of sphere center depth are shown in Figure 6. The present numerical results are in good agreements with reference results, especially with the experimental ones.

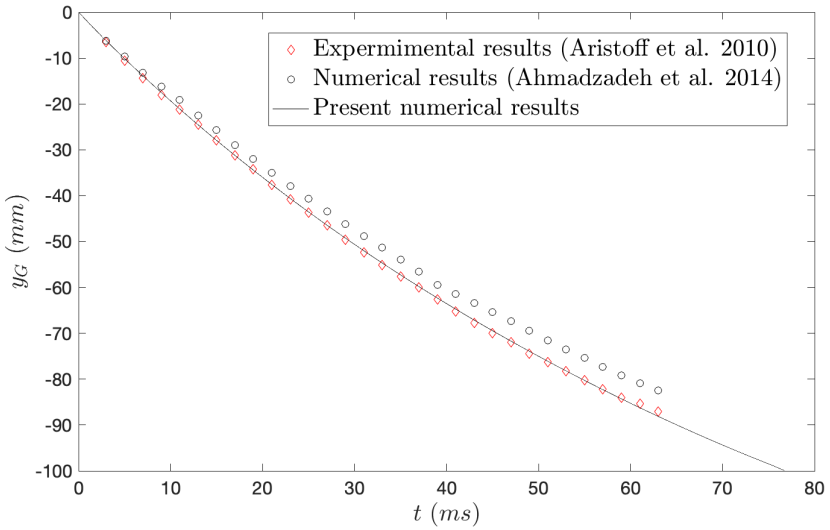


Figure 6. Temporal evolution of the vertical position y_G of the center of mass of the sphere.

3. Numerical results

3.1. Numerical configuration and parameters

The densities and dynamic viscosities of the fluids are $\rho^+ = 1000\,kg/m^3$ and $\mu^+ = 10^{-3}\,Pa.s$ for the water, and $\rho^- = 1.2\,kg/m^3$ and $1.87 \cdot 10^{-5}\,Pa.s$ for the air. The surface tension is $\sigma = 0.0728\,N/m$.

The dolphin geometry is extracted from the supplementary material from (Tanaka et al. 2019) and the geometry is reproduced in Figure 2. Here, the length of the dolphin is imposed to be $\ell = 3\,m$, leading to total mass $m = 316\,kg$ with $\rho_S = 1000\,kg/m^3$.

The computational domain under consideration is $x \in [-12, 4]\,m$, $y \in [-3, 9]\,m$ and $z \in [-2, 2]\,m$. At rest, the air-water interface Γ_f is located at $y_I = 4.5\,m$, and the center of mass of the dolphin is $\boldsymbol{x}_G = (x_G,\, y_G,\, z_G) = (0,\, 0,\, 0)\,m$. The initial orientation of the dolphin is $\boldsymbol{\theta} = (\theta_x,\, \theta_y,\, \theta_z) = (0,\, 0,\, \theta_0)$. While the initial angular velocity is $\boldsymbol{\omega} = \mathbf{0}$, the initial dolphin linear velocity is $\overline{\boldsymbol{u}} = (u_G,\, v_G,\, w_G) = (0,\, v_0,\, 0)$, where v_0 can be a positive constant to decrease the distance to reach a sufficient velocity for a real jump, and thus to decrease the size of the computational domain.

The velocity boundary condition on the external boundaries Γ_{ext} are the following. On the side boundaries (left, right, front, back), periodic conditions are imposed, $\boldsymbol{u}_{front} = \boldsymbol{u}_{back}$ and $\boldsymbol{u}_{left} = \boldsymbol{u}_{right}$. No slip boundary conditions are imposed on the bottom $\boldsymbol{u}_{bottom} = (0,\, 0,\, 0)$, and streamline type of boundary conditions are imposed on the top, *i.e.* $v_{top} = 0$, $\dfrac{\partial u_{top}}{\partial y} = \dfrac{\partial w_{top}}{\partial y} = 0$.

The dolphin surface is discretized with a regular mesh composed by $200 \times 200 = 40,\,000$ cells, see two bottom figures in 2. Even if large skewness of mesh near fins and

tails are observed, the mesh is a tessellation of the body surface on which integrals (30) and (31) can be discretized. This mesh is similar to those used in (Bergmann & Iollo 2016, Bergmann, Iollo & Mittal 2014) for carangiform and thunniform swimmers.

The computational domain is discretized by $640 \times 480 \times 160 \approx 50 \cdot 10^6$ cells, and the step size is $h = \Delta x = \Delta y = \Delta z = 0.025\,m$. The time step is defined with a CFL condition equal to 0.3.

Due to the large dolphin deformations, boundary layers are detached and it is thus not necessary to use a costly boundary layer mesh. The turbulence model introduced in (Vreman 2004) is used. Other turbulence model may be used to study the laminar-turbulence transition (Riedeberger & Rist 2012).

Artificial waves at the air-water interface are generated by a Kelvin-Helmholtz instability using initial conditions $u(\boldsymbol{x}, 0) = 0\,m/s$ in the water ($\boldsymbol{x} \in \Omega_f^+$) and $u(\boldsymbol{x}, \boldsymbol{0}) \neq 0\,m/s$ in the air ($\boldsymbol{x} \in \Omega_f^-$).

From the initial water surface position at $y_I = 4.5\,m$, we define two other locations: the first one is over water, $y_I^+ = y_I + 0.6\,m$, and the second one is under water, $y_I^- = y_I - 0.8\,m$. We denote by $T = 0.2\,s$ a transition period over that p goes linearly from 0 to 1 or from 1 to 0.

The dolphin body deformation is defined according to the following swimming law.

(i) In order to produce enough propulsive effect to get out of the water starting only $4.5\,m$ under the surface, we imposed a sinusoidal law with large amplitude and frequency, *i.e.* $A = 1.5\,m$ and $f = \omega/(2\pi) = 4\,Hz$, with initial linear transition with $p(0) = 0$ and $p(T) = 1$. This swimming law is performed until the dolphin center of mass y_G reaches y_I^+ at time t_I^+, and we thus have $p(t) = 1\ \forall t \in [T,\, t_I^+]$. The periodic swimming law is then linearly damped to reach $p(t_I^+ + T) = 0$.

(ii) When the periodic swimming is damped from $t = t_I^+$, a mean curvature $\overline{\kappa} = 1/R$ with $R = 2\,m$ is progressively added to the whole midline. The smoothing function q depends on the vertical position of the center of mass such that maximal curvature is obtained when the dolphin is in the air. When $y_G = y_I$, at the end of the jump, the function q is damped and the curvature tends to zero.

(iii) When the dolphin re-enter into the water due to the gravity, more precisely when $y_G < y_I^-$ at $t = t_I^-$, another periodic swimming with $A = 1\,m$ and $f = 2\,Hz$ is activated with a linear transition with $p(t_I^-) = 0$ and $p(t) = 1$ for $t \geq t_I^- + T$.

Mathematically, the function p depends on some temporal intervals defined by:

$$p(t) = \begin{cases} t/T & \text{if } 0 \leq t < T, \\ 1 & \text{if } T \leq t < t_I^+, \\ (t_I^+ + T - t)/T & \text{if } t_I^+ \leq t < t_I^+ + T, \\ 0 & \text{if } t_I^+ + T \leq t < t_I^-, \\ (t - t_I^-)/T & \text{if } t_I^- \leq t < t_I^- + T, \\ 1 & \text{if } t \geq t_I^- + T, \end{cases} \tag{32}$$

and the function q is linked to the position y_G of the dolphin center of mass for $t \leq t_I^-$:

$$q(t) = \frac{1}{\max\left(1, \dfrac{1}{(10^{-4} + \text{minmod}\{0.7(t - t_I^+),\, 1\}\,(y_G(t) - y_I^-))^2}\right)}, \tag{33}$$

where

$$\text{minmod}\{a,\, b\} = \begin{cases} a & \text{if } |a| < |b|,\ ab > 0 \\ b & \text{if } |b| < |a|,\ ab > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{34}$$

This handcrafted law has been developed from several numerical experiments to mimic as best as possible the dolphin jump.

3.2. Numerical results

A numerical simulation with $\theta_0 = 60°$ and $V = 4\,m/s$ is performed using 256 processors. The initial air velocity is $u(\boldsymbol{x},\, 0) = -5\,m/s$ for $(x) \in \Omega_f^-$.

The temporal evolution of functions p and q are given in figure 7(b), where the different time t_I^-, t_I, t_I^+ and the other ones necessary to defined functions p and q are plotted using vertical lines. These lines depend on the vertical position y_G of the dolphin center of mass following (32) and (33), see figure 7(a). The periods with $p(t) \geq 0$ are highlighted in blue, and the period where $q(t) \geq 0$ is highlighted in red. Blank zones thus denote undeformed body shape.

Figure 8 shows the temporal evolution of the linear and angular position of the dolphin, the linear velocity, and the forces and torques exerted by the fluid on the body. The vertical velocity of the dolphin goes from $v_G = 4\,m/s$ at $t = 0\,s$ to $v_G = 6.4\,m/s$ at $t = t_I^+$, see figure 8(b). Note that we have chosen an initial velocity $v_G > 0$ to make use of a smaller computational domain. After this initial acceleration, the obtained velocity allows the dolphin to go out of water (figure 8(a)). When the dolphin is the air (in the red zone in figure 8), the dolphin vertical velocity v_G and the angle θ_z start to decrease due to gravity effect and the curvature. After re-entry into water, a new sinusoidal swimming law is performed and the the dolphin swims in the water with a position $\theta_z \approx -10°$ along which a slight acceleration is observed.

Eight snapshots $\{S_i\}_{i=1,..8}$ uniformly extracted during the actual numerical simulation are presented in figures 9, 10 and 11. These snapshots correspond to $\{t_i\}_{i=1,..8}$ with $t_i = 0.45\,(i-1) + 0.6\,s$ and are plotted in figure 8 using vertical black lines. While figures 9 and 10 present the dolphin jump visualisation of the water surface from different views, figure 11 presents snapshots of the Q-crietrion.

The two first snapshots S_1 and S_2 correspond to the first periodic swimming law for propulsion. Snapshots S_3 and S_4 correspond to a period in the air where maximal curvature $\overline{\kappa}$ is imposed. The fifth snapshot, S_5 is extracted for almost undeformed body shape when re-entering in water. This deformation corresponds to $q\,\overline{\kappa} = 0.007\,m^{-1}$ and

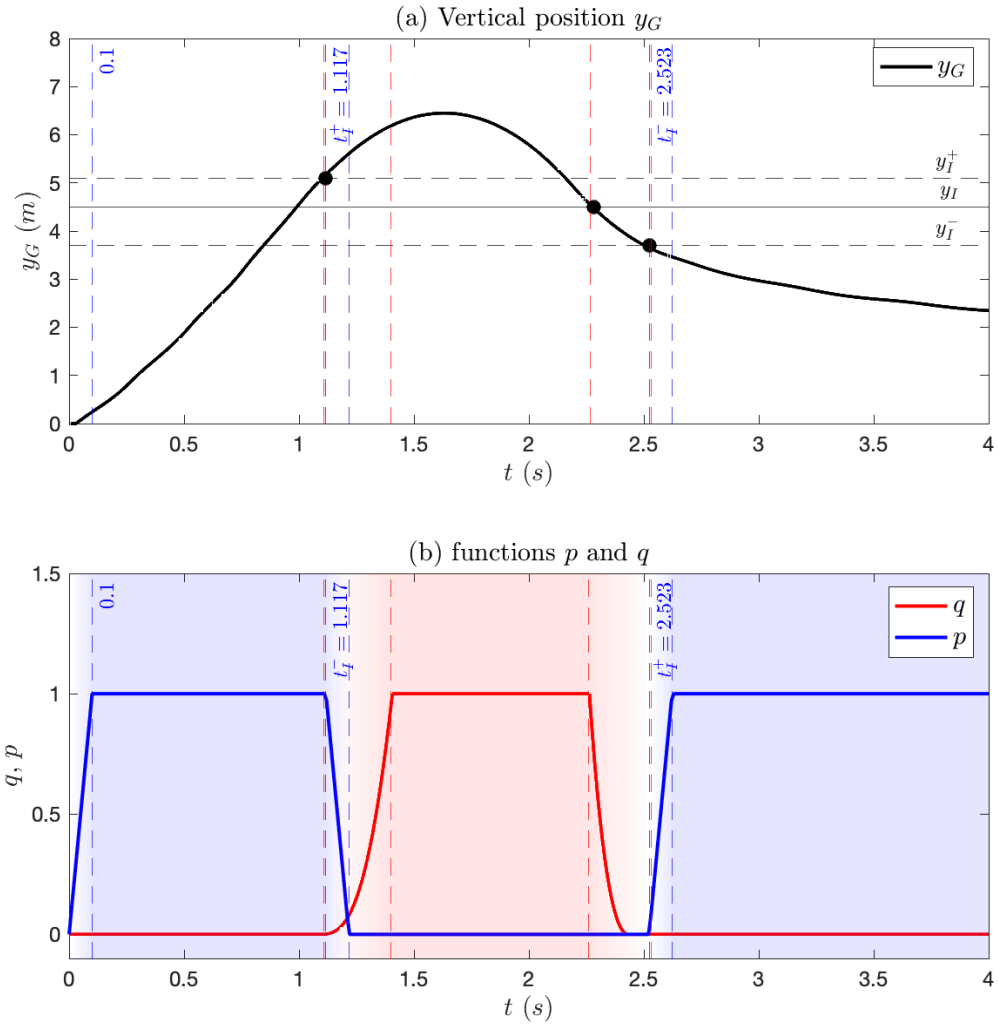


Figure 7. Temporal evolution of functions p and q and the vertical position of the dolphin center of mass y_G.

$p = 0$. Finally, snapshots S_6, S_7 and S_8 correspond to the second periodic swimming law when the dolphin is back in water. In water, the vortices generated by the self propelled dolphin are similar to whose observed for most of fish like swimmers (Zhu et al. 2002, Triantafyllou et al. 2000, Liao et al. 2003, Bergmann & Iollo 2016). Other kind of vortices are generated in the shear layer at the water-air interface. Indeed, the initial water and air velocity are respectively $u = 0\,m/s$ and $u = -5\,m/s$.

Trajectories of the dolphin center of mass corresponding to different initial velocity v_0 and angle θ_0 are presented in figure 12. As expected, the highest jumps correspond to highest initial velocities. The correlation between initial velocity and the jump can not be easily determined. Indeed, the periodic swimming law is not stopped at the

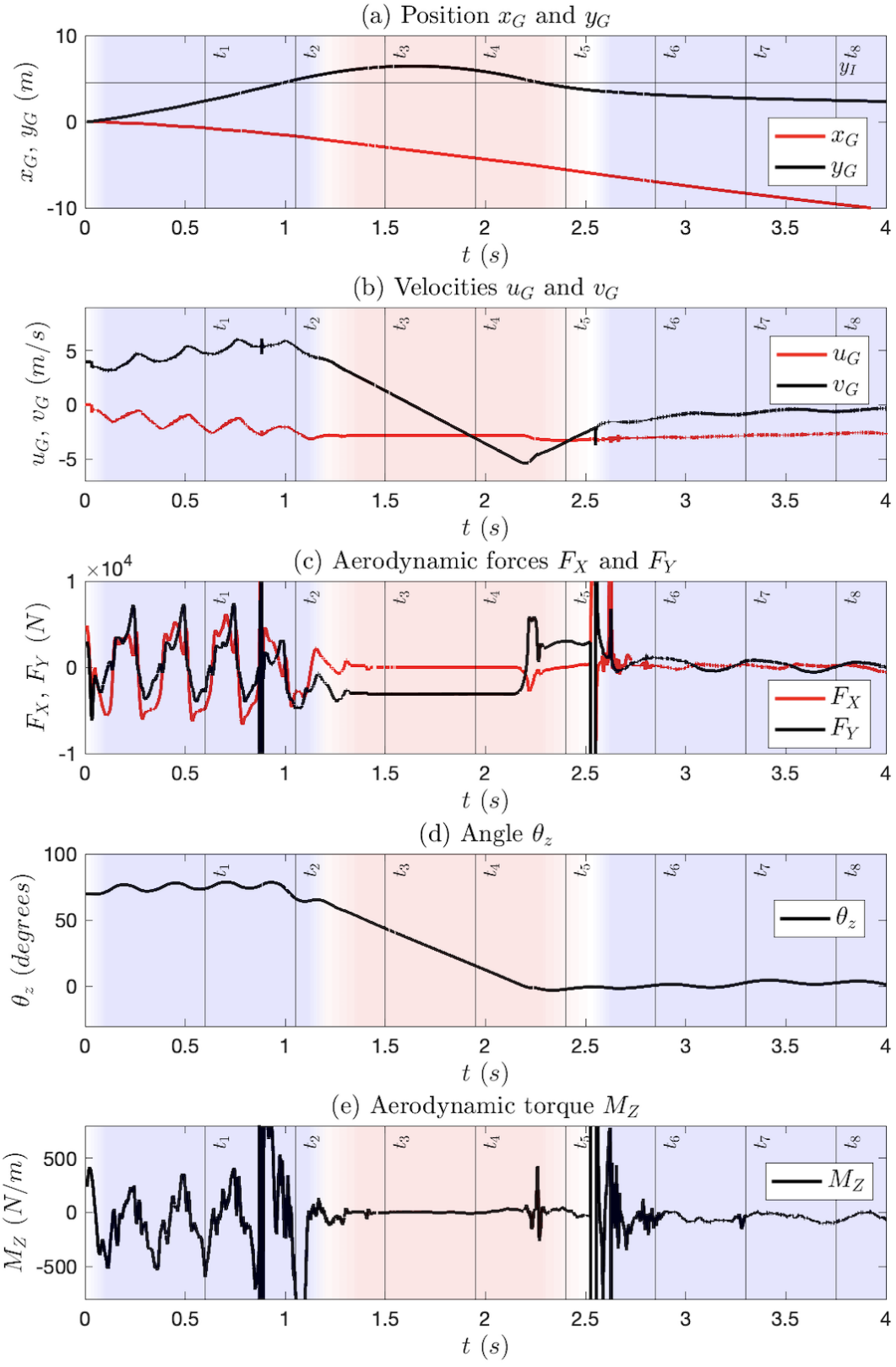
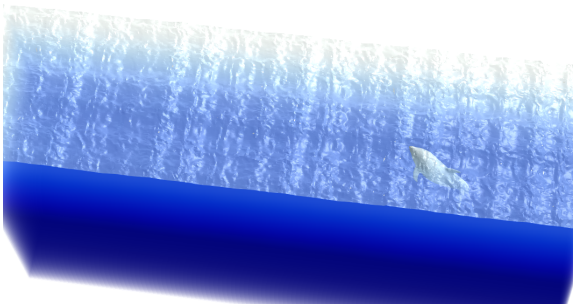
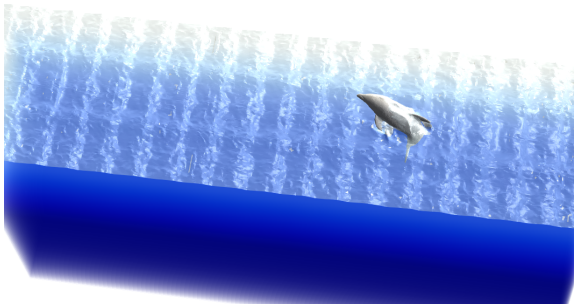


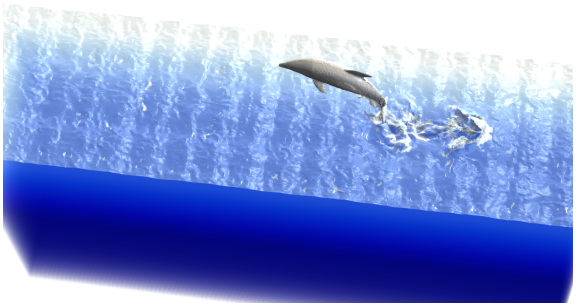
Figure 8. Temporal evolutions of dolphin linear and angular position and velocity, and aerodynamic forces and torques.



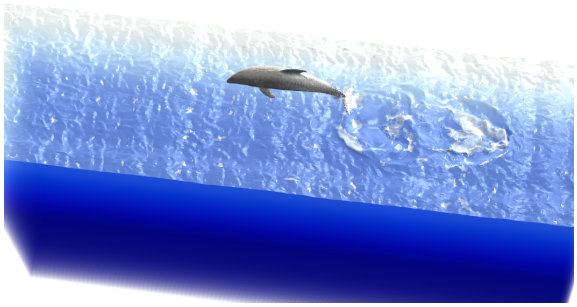
(a) $t_1 = 0.60\,s$.



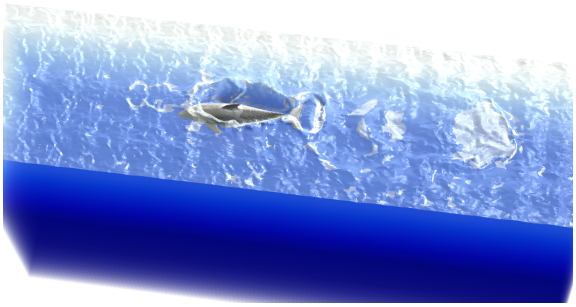
(b) $t_2 = 1.05\,s$.



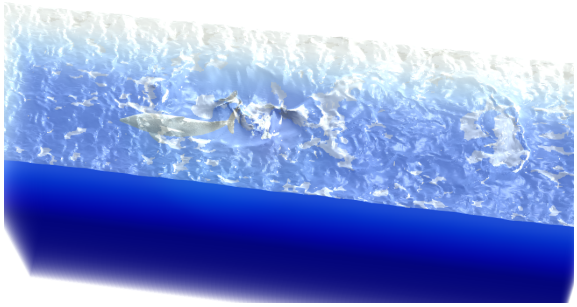
(c) $t_3 = 1.50\,s$.



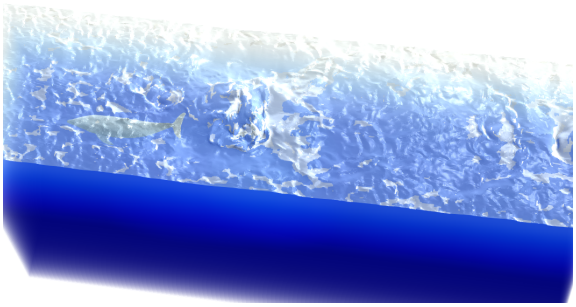
(d) $t_4 = 1.95\,s$.



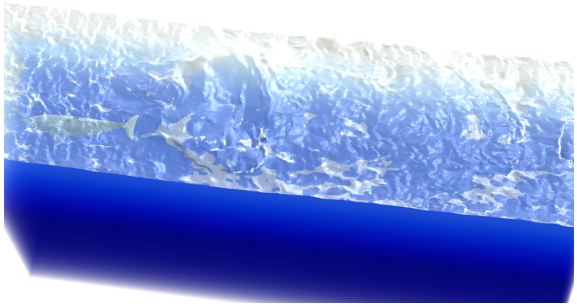
(e) $t_5 = 2.40\,s$.



(f) $t_6 = 2.85\,s$.

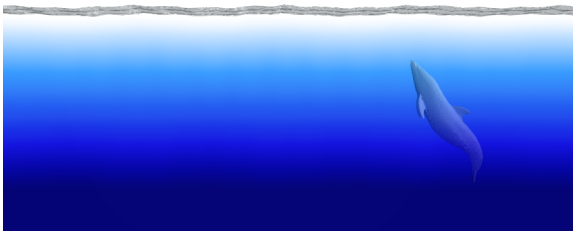


(g) $t_7 = 3.30\,s$.

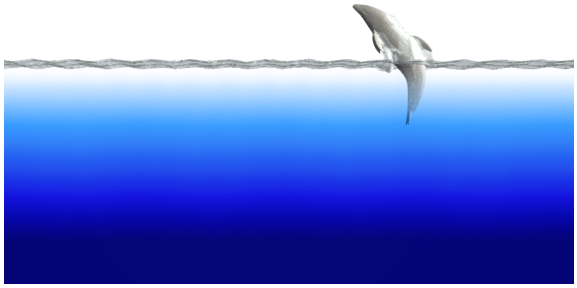


(h) $t_8 = 3.75\,s$.

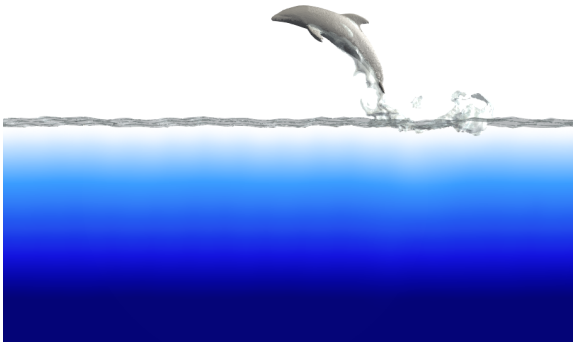
Figure 9. Snapshots of the dolphin with water surface for $\theta_0 = 60°$ and $v_0 = 4\,m/s$. Top view.



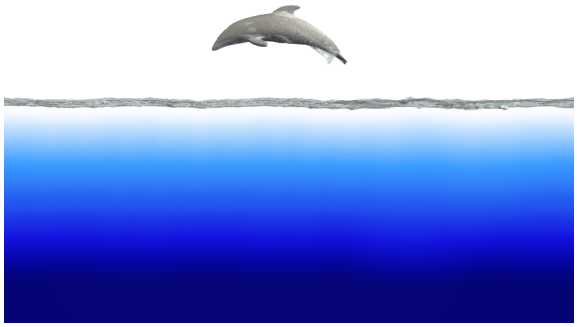
(a) $t_1 = 0.60\,s$.



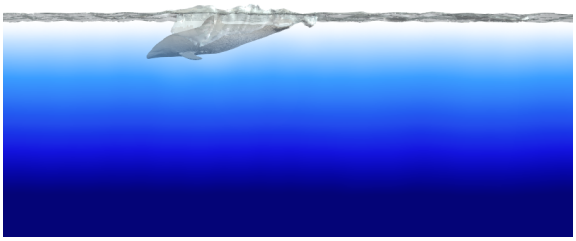
(b) $t_2 = 1.05\,s$.



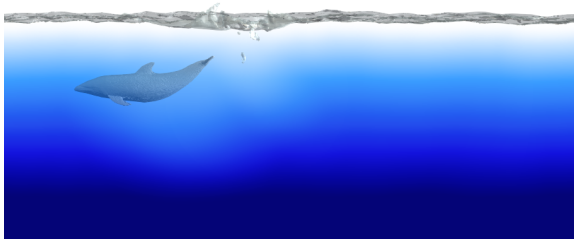
(c) $t_3 = 1.50\,s$.



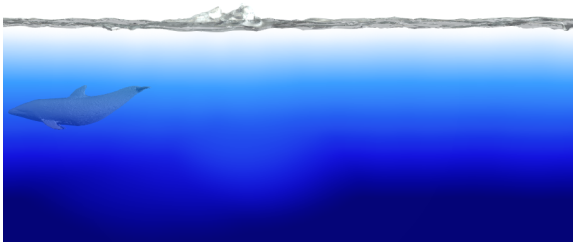
(d) $t_4 = 1.95\,s$.



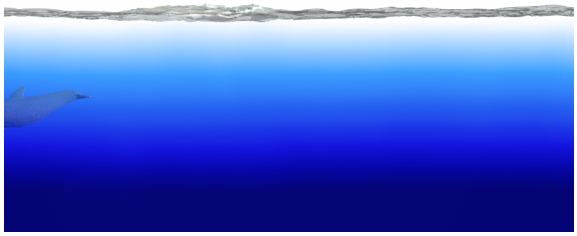
(e) $t_5 = 2.40\,s$.



(f) $t_6 = 2.85\,s$.

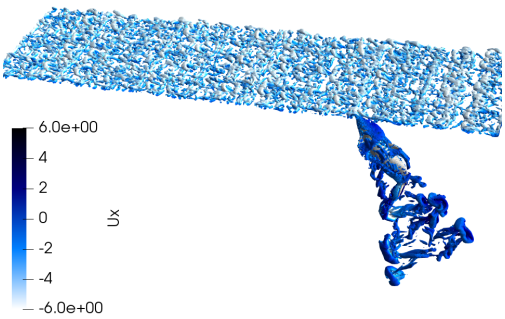


(g) $t_7 = 3.30\,s$.

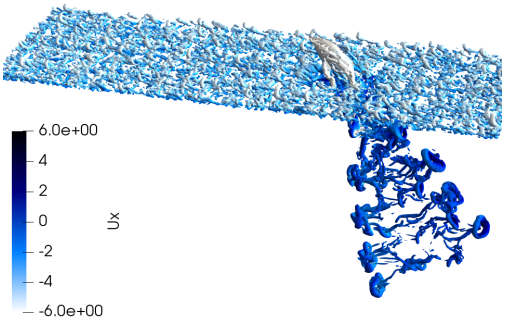


(h) $t_8 = 3.75\,s$.

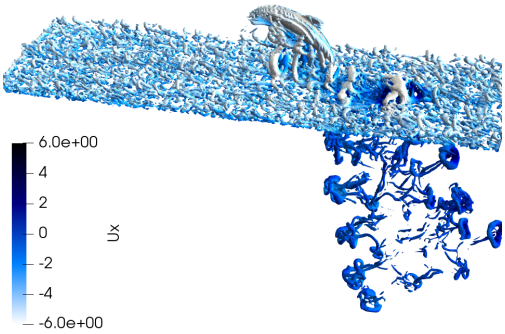
Figure 10. Snapshots of the dolphin with water surface for $\theta_0 = 60°$ and $v_0 = 4\,m/s$. Front view.



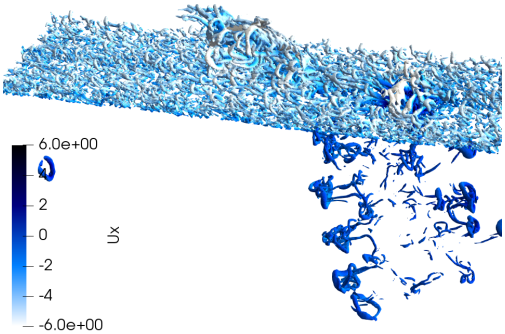
(a) $t_1 = 0.60\,s$.



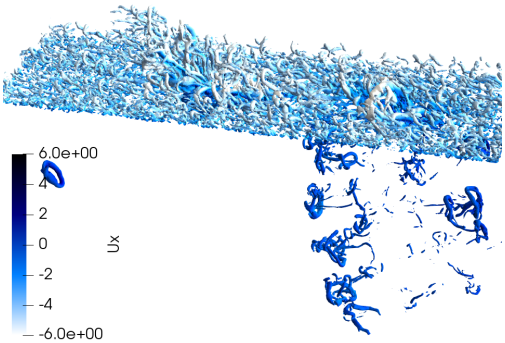
(b) $t_2 = 1.05\,s$.



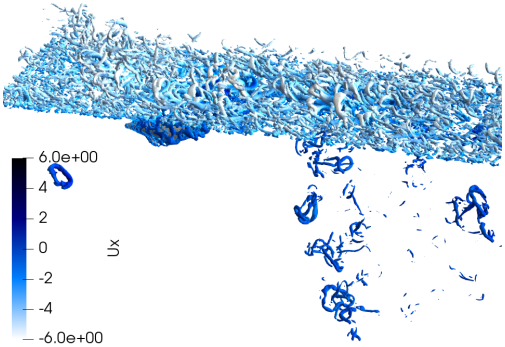
(c) $t_3 = 1.50\,s$.



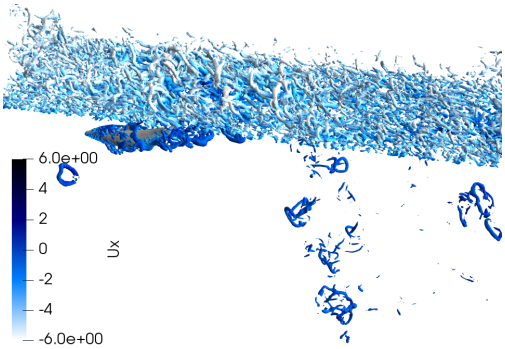
(d) $t_4 = 1.95\,s$.



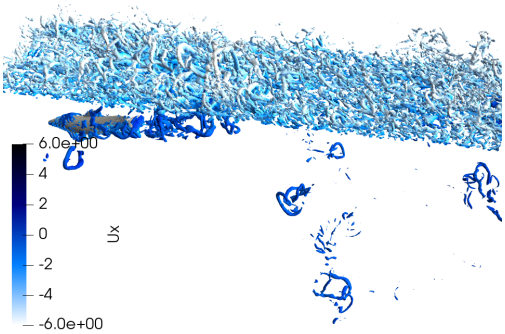
(e) $t_5 = 2.40\,s$.



(f) $t_6 = 2.85\,s$.



(g) $t_7 = 3.30\,s$.



(h) $t_8 = 3.75\,s$.

Figure 11. Snapshots of the dolphin with water surface for $\theta_0 = 60°$ and $v_0 = 4\,m/s$. Front view.

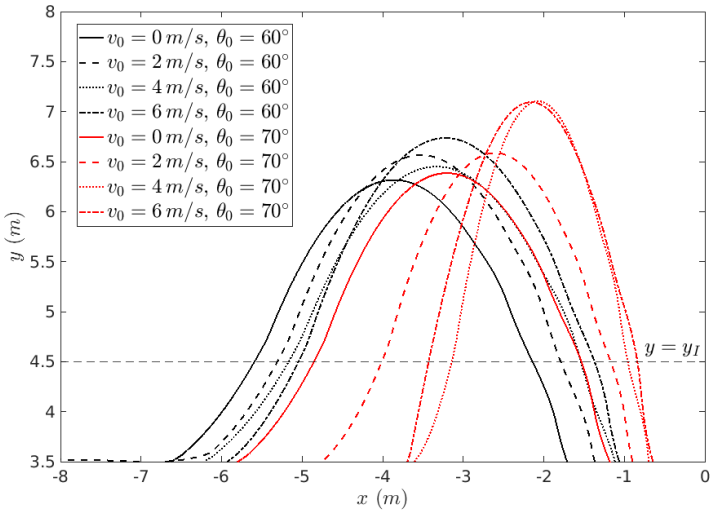


Figure 12. Comparison of the dolphin jump with different initial conditions v_0 and θ_0. Jump is from right to left.

same time during the swimming stroke, thus influencing the jump velocity and angle at $y = y_I$.

In what follows we compare the dolphin jump with pure ballistic trajectories. The ballistic equations are:

$$\begin{cases} x_b(t) = x_0 + \overline{v}_0\, t\, cos(\alpha), \\ y_b(t) = y_0 + \overline{v}_0\, t\, sin(\alpha) - \dfrac{1}{2}\, g\, t^2, \end{cases} \tag{35}$$

where $(x_0,\, y_0)$ is initial projectile position, α is the initial trajectory orientation at $(x_0,\, y_0)$, $\overline{v}_0$ is the initial velocity along the α direction at $(x_0,\, y_0)$, $g = 9.81\, m/s^2$ is the gravity acceleration and t is the time. Figure 13 shows the real dolphin trajectory and several ballistic trajectories corresponding to different initial positions $(x_0,\, y_0)$.

For $y_0 = y_I$ at $t = 0.944\, s$, *i.e.* when the dolphin center of mass reach the water surface, large deviations between the dolphin and ballistic trajectories are observed. This is due to the fact that the dolphin is still accelerating (see figure 8). For $y_0 = y_I^+ = y_I + 0.6\, m$ at $t = 1.117\, s$, the periodic swimming law is progressively stopped. A part of the dolphin is still in water, the acceleration is positive, and the deviation between the real trajectory and the ballistic trajectory is lower. Finally, for $y_0 = y_I + 0.8\, m$ at $t = 1.23\, s$ the acceleration tends slowly to zero and the ballistic trajectory almost perfectly matches the real dolphin trajectory, until the dolphin goes back to water at $y_G = y_I$. This is an obvious observation since the forces exerted by the air on the dolphin ($F_X \approx 6\, N$, see figure 8) are quite low comparing to the inertial effect of a $318\, kg$ dolphin with velocity around $6\, m/s$.

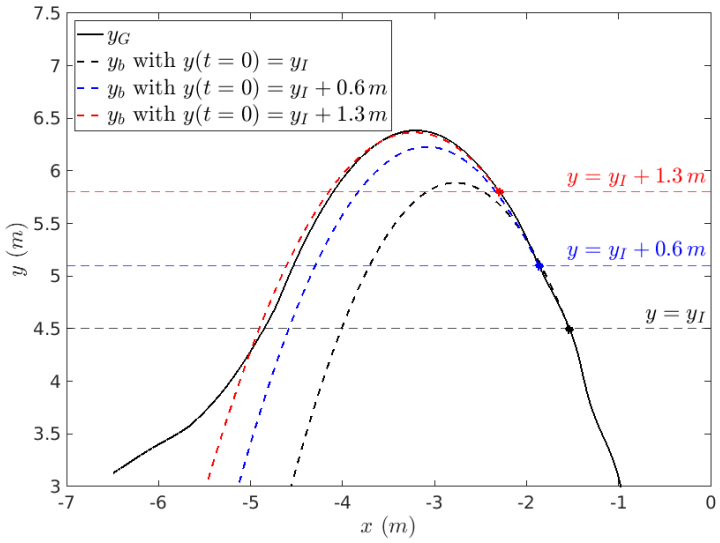


Figure 13. Comparison of the dolphin jump ($v_0 = 4\,m/s$ and $\theta_0 = 70°$) with ballistic trajectories. Jump is from right to left.

4. Conclusions

In this paper we have developed a numerical model of the dolphin jump. This model deals with complex physical phenomena such as fluid-structure interactions and liquid interface involving large deformations with possible topological changes (bubbles and droplets). The numerical model relies on fictitious domain approaches where the mesh used to solve the fluid does not follow the interfaces. The CSF method is used to model the water-air interface, and the Volume Penalization method is used to take into account the dolphin's body. The numerical results are qualitatively in good agreements with the dolphin jumps observed in nature. The dolphin has to generate large forces to reach a velocity threshold necessary for the jump. The jump is similar to a ballistic projectile where the resistive forces are neglected. This hypothesis is justified since the computed resistive forces are only around 6 Newtons for a $320\,kg$ dolphin with a velocity around $6\,m/s$. We are now working on the optimization of the initial periodic swimming law to generate maximal acceleration to produce the highest jump.

Acknowledgements

Computer time for this study was provided by the computing facilities MCIA (Mésocentre de Calcul Intensif Aquitain) of the Université de Bordeaux and of the Université de Pau et des Pays de l'Adour.

References

Adalsteinsson D & Sethian J 1999 *Journal of Computational Physics* **148**(1), 2–22.

Ahmadzadeh M, Saranjam B, Hoseini Fard A & Binesh A 2014 *Applied Mathematical Modelling* **38**(5), 1673–1684.
URL: *https://www.sciencedirect.com/science/article/pii/S0307904X13005647*

Ahmed S, Bak S, McLaughlin J & Renzi D 2011 *SIAM Journal on Scientific Computing* **33**(5), 2402–2420.

Angot P, Bruneau C & Fabrie P 1999 *Num. Math.* **81**(4), 497–520.

Aristoff J M, Truscott T T, Techet A H & Bush J W M 2010 *Physics of Fluids* **22**(3), 032102.
URL: *https://doi.org/10.1063/1.3309454*

Bale R, Hao M, Bhalla A, Patel N & Patankar N 2014 *Scientific reports* **4**, 5904.

Barrett D, Triantafyllou M, Yue D, Grosenbauch M & Wolfgang M 1999 *J. Fluid Mech.* **392**, 182–212.

Beal D, Hover F, Triantafyllou M, Liao J & G.V. L 2006 *J. Fluid Mech.* **549**, 385–402.

Bergmann M, Cordier L & Brancher J P 2006 *Phys. Fluids* **18**(2), 028101:1–4.

Bergmann M, Hovnanian J & Iollo A 2014 *Communications in Computational Physics* **15**(5), 1266–1290.

Bergmann M & Iollo A 2011 *Journal of Computational Physics* **230**(2), 329 – 348.

Bergmann M & Iollo A 2016 *Journal of Computational Physics* **323**, 310 – 321.
URL: *http://www.sciencedirect.com/science/article/pii/S0021999116303175*

Bergmann M, Iollo A & Mittal R 2014 *Bioinspiration & Biomimetics* **9**(4), 046001.
URL: *http://stacks.iop.org/1748-3190/9/i=4/a=046001*

Bhalla A P S, Bale R, Griffith B E & Patankar N A 2014 *Journal of Computational Physics* **256**, 88–108.

Bhalla A P S, Griffith B E & Patankar N A 2013 *PLOS Computational Biology* **9**(6), 1–16.
URL: *https://doi.org/10.1371/journal.pcbi.1003097*

Bozkurttas M, Tangorra J, Lauder G & Mittal R 2009 *in* 'Mining Smartness from Nature (CIMTEC 2008)' Vol. 58 of *Advances in Science and Technology* Trans Tech Publications Ltd pp. 193–202.

Brackbill J, Kothe D & Zemach C 1992 *Journal of Computational Physics* **100**(2), 335 – 354.

Chang B, Myeong J, Virot E, Clanet C, Kim H Y & Jung S 2019 *Journal of The Royal Society Interface* **16**(152), 20190014.
URL: *https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2019.0014*

Chorin A 1968 *Math. Comp.* **22**, 745–762.

Cox R 1986 *Journal of Fluid Mechanics* **168**, 169–194.

Fish F E, Nicastro A J & Weihs D 2006 *Journal of Experimental Biology* **209**(4), 590–598.
URL: *https://doi.org/10.1242/jeb.02034*

Fish F & Lauder G 2006 *Annu. Rev. Fluid Mech.* **38**, 193–224.

Gao A & Triantafyllou M S 2018 *Journal of Fluid Mechanics* **850**, 304–335.

Gopalkrishnan R, Triantafyllou M S, Triantafyllou G S & Barrett D 1994 *Journal of Fluid Mechanics* **274**, 1–21.

Gray J 1936 *J. Exp. Biol.* **13**, 192–199.

Han P, Wang J, Fish F & Dong H 2020.

Jiang G S & Shu C W 1996 *Journal of Computational Physics* **126**(1), 202 – 228.

Kern S & Koumoutsakos P 2006 *The Journal of Experimental Biology* **209**, 4841–4857.

Li G, Kolomenskiy D, Liu H, Thiria B & Godoy-Diana R 2022 *Frontiers in Robotics and AI* **9**.
URL: *https://www.frontiersin.org/article/10.3389/frobt.2022.825889*

Liao J, Beal D, Lauder G & Triantafyllou M 2003 *Science* **302**, 1566–1569.

Lighthill M 1970 *J. Fluid Mech.* **44**, 265–301.

Liu X D, Osher S & Chan T 1994 *Journal of Computational Physics* **115**(1), 200 – 212.

Luddens F, Bergmann M & Weynans L 2015 *International Journal for Numerical Methods in Fluids* **79**, 654–675.

Maertens A P, Gao A & Triantafyllou M S 2017 *Journal of Fluid Mechanics* **813**, 301–345.

Mendelson L & Techet A H 2020 *Bioinspiration & Biomimetics* **16**(1), 016006.
URL: *https://doi.org/10.1088/1748-3190/abb78e*

Mittal R, Dong H, Bozkurttas M, Najjar F, Vargas A & von Loebbecke A 2008 *Journal of Computational*

Physics **227**(10), 4825 – 4852.

Mittal R & Iaccarino G 2005 *Annu. Rev. Fluid Mech.* **37**, 239–261.

Nangia N, Griffith B E, Patankar N A & Bhalla A P S 2019 *Journal of Computational Physics* **390**, 548–594.
URL: *https://www.sciencedirect.com/science/article/pii/S0021999119302256*

Nangia N, Johansen H, Patankar N A & Bhalla A P S 2017 *Journal of Computational Physics* **347**, 437 – 462.
URL: *http://www.sciencedirect.com/science/article/pii/S0021999117305016*

Nangia N, Patankar N A & Bhalla A P S 2019 *Journal of Computational Physics* **398**, 108804.
URL: *https://www.sciencedirect.com/science/article/pii/S0021999119304887*

Osher S & Sethian J A 1988 *J. Comput. Phys.* **79**(12).

Patankar N A 2003 *Langmuir* **19**(4), 1249–1253.
URL: *https://doi.org/10.1021/la026612+*

Pavlov V 2006 *Bioinspiration & biomimetics* **1**, 31–40.

Peskin C 1972 *J. Comp. Phys.* **10**, 252–275.

Riedeberger D & Rist U 2012 *in* W. E Nagel, D. B Kröner & M. M Resch, eds, 'High Performance Computing in Science and Engineering '11' Springer Berlin Heidelberg Berlin, Heidelberg pp. 379–391.

Russo G & Smereka P 2000 *Journal of Computational Physics* **163**(1), 51–67.

Sethian J 1996 *Applied Mathematics* **93**, 1591–1595.

Sethian J A 1999 *Level Set Methods and Fast Marching Methods* Cambridge University Press, Cambridge, UK.

Shirgaonkar A A, MacIver M A & Patankar N A 2009 *Journal of Computational Physics* **228**(7), 2366 – 2390.

Smits A J 2019 *Journal of Fluid Mechanics* **874**.

Sussman M, Smereka P & Osher S 1994 *Journal of Computational Physics* **114**(1), 146 – 159.

Tanaka H, Li G, Uchida Y, Nakamura M, Ikeda T & Liu H 2019 *PLOS ONE* **14**(1), 1–25.
URL: *https://doi.org/10.1371/journal.pone.0210860*

Temam R 1969 *Archiv. Rat. Mech. Anal.* **32**, 377–385.

Triantafyllou M S, Techet A H, Zhu Q, Beal D N, Hover F S & Yue D K P 2002 *Integrative and Comparative Biology* **42**(5), 1026–1031.
URL: *https://doi.org/10.1093/icb/42.5.1026*

Triantafyllou M, Triantafyllou G & Yue D 2000 *Annual Review of Fluid Mechanics* **32**.

Truscott T T, Epps B P & Belden J 2014 *Annual Review of Fluid Mechanics* **46**(1), 355–378.
URL: *https://doi.org/10.1146/annurev-fluid-011212-140753*

von Loebbecke A, Mittal R, Fish F & Mark R 2009*a* *Human Movement Science* **28**(1), 99–112.
URL: *https://www.sciencedirect.com/science/article/pii/S016794570800081X*

von Loebbecke A, Mittal R, Fish F & Mark R 2009*b* *J Biomech Eng.* **131**(5), 054504.

von Loebbecke A, Mittal R, Mark R & Hahn J 2009 *Sports biomechanics / International Society of Biomechanics in Sports* **8**, 60–77.

Vreman A 2004 *Physics of Fluids* **16**(10), 3670–3681.

Zhu Q, Wolfgang M, Yue D & Triantafyllou M 2002 *J. Fluid Mech.* **468**, 1–28.