Physics-informed Reinforcement Learning for Perception and Reasoning about Fluids

Beatriz Moya\textsuperscript{1}, Alberto Badias\textsuperscript{2}, David Gonzalez\textsuperscript{1}, Francisco Chinesta\textsuperscript{3}, Elias Cueto\textsuperscript{1}

\textsuperscript{1}Aragon institute in Engineering Research. University of Zaragoza, Spain
\{beam, gonzal, ecueto\}@unizar.es

\textsuperscript{2}Polytechnic University of Madrid, Spain
alberto.badias@upm.es

\textsuperscript{3}ESI Group chair. PIMM Lab, ENSAM Institute of Technology, France
francisco.chinesta@ensam.eu

Abstract

Learning and reasoning about physical phenomena is still a challenge in robotics development, and computational sciences play a capital role in the search for accurate methods able to provide explanations for past events and rigorous forecasts of future situations. We propose a physics-informed reinforcement learning strategy for fluid perception and reasoning from observations. As a model problem, we take the sloshing phenomena of different fluids contained in a glass. Starting from full-field and high-resolution synthetic data for a particular fluid, we develop a method for the tracking (perception) and analysis (reasoning) of any previously unseen liquid whose free surface is observed with a commodity camera. This approach demonstrates the importance of physics and knowledge not only in data-driven (grey box) modeling but also in the correction for real physics adaptation in low data regimes and partial observations of the dynamics. The method here presented is extensible to other domains such as the development of cognitive digital twins, able to learn from observation of phenomena for which they have not been trained explicitly.

1 Introduction

The community has witnessed great advances in deep learning and Artificial Intelligence (AI) concerning the imitation of human-like skills [37] [49]. However, despite experiencing great flourishing, works are still transitioning to the development of the so-called Artificial General Intelligence (AGI) [17]. In this next step of AI, we approach more anthropomorphic skills, independence and high-level reasoning for decision-making [57] and behavior learning [2] [36]. One of the requirements to cross the boundary between AI to AGI is to improve the development of sensory and physics perception in machine systems. Physics perception refers to the interpretation and reasoning skills that interpret sensed data. These technologies play a capital role in robotics, especially for planning and control tasks [49]. For this purpose, we need interactive, real-time simulators of real-world physics, constructed by a judicious mixing of theory, data, and computation [37].

Reinforcement learning (RL) constitutes an interesting framework for the construction of perception strategies from the interaction with the medium [51]. Its main purpose is to reward those actions that lead to a bound objective. Reinforcement learning can also be applied to the optimization of learned simulators of complex dynamical systems, thus mimicking the way we, humans, learn intuitive representations of the physical world that surrounds us.

Model-based reinforcement learning [54] relies on models of the physical world when the collection of data is difficult, or the required information is inaccessible [31]. This information is thus completed by a model that recreates the perception of real physics to make decisions about future events. Given the high complexity of the physics involved in real events, RL optimization usually relies on data-inspired modeling techniques of the dynamics to achieve high adaptivity and perform the optimization of the model and control policies all at once. Works include examples based on linear regression [44], the use of diverse neural networks [29] [43] and Gaussian processes [12]. Con-
versely, current proposals introduce physics-informed machine learning into the model’s inference [38]. Among the plethora of existing approaches, knowledge distillation is mainly accomplished by learning PDEs’ parameters [46] [4], the imposition of Hamiltonian or Lagrangian structures [9] [59] [14], or thermodynamics-informed simulators [39] [56] [58] [30].

Although physics-informed simulators reach higher generalization than unconstrained models, there are still difficulties to match the model outputs with an evolving real physical environment [3]. For this reason, the learned simulator can be formulated as a reinforcement learning optimization problem to adapt to new scenarios through continuous observation. In addition, the imposition of inductive biases could be found advantageous in the correction process to reduce error bounds and adapt to new scenarios more efficiently [18]. Examples of model discovery with reinforcement learning include the Burger’s Equation as a common benchmark [8] [5] [55], or the Kuramoto-Sivashinsky Equation [10]. In this context, this strategy also targets turbulence and flow control problems, where the object of the optimization is also the physics simulator [45] [47] [53] [16] [42].

In this work, we develop a method for the perception (tracking of the free surface of the fluid) and reasoning (providing the user with full-field information—velocity, stress, ...) about the physical state of a sloshing fluid. While previous works employ simulation as the engine of physical scene understanding (see [1] or [7], for instance), thus needing previous knowledge on the physics of the scene and a pre-defined simulator, our method constructs on the fly a learned simulator. This learned simulator is built by resorting initially to synthetic full-field data, possibly coming from different fluids. It then makes use of reinforcement learning in a thermodynamics-informed setting to correct systematic deviations of the observed reality from its predictions. Our approach ensures the compliance to first principles—conservation of energy, non-negative entropy production—of the resulting simulations, even if they are constructed from partial observations of the reality (in our case, the observation of the free surface of the fluid).

Both the originally learned simulator and its subsequent reinforcement learning optimization are based on the use of the General Equation for the Non-Equilibrium Reversible-Irreversible Coupling (GENERIC) formalism as an inductive bias to guide the training by the laws of the thermodynamics [23]. The physical knowledge already gained in the source model is to be preserved for the sake of the physical interpretability of the results and the success of the adaptation. The proposed thermodynamics-informed scheme will adapt to the inherent dissipative nature of real phenomena. The model is to be corrected from the sole observation of the free surface of the new liquid. As a result, we obtain a corrected, augmented intelligence system that performs the integration of the fluid dynamics in real-time from the evaluation of the free surface evolution. The output is the new state of the fluid integrated in time. This approach is coupled with a computer vision system to build a closed-loop algorithm that learns liquid slosh from real measurements. The adaptation is performed in an off-line phase to prize the cumulative reward over some acquired data. After the correction, the simulation of the dynamics is accomplished in real-time, obtaining improved recreations of the real slosh.

The development of this method revolves around the convergence of the aforementioned reinforcement learning strategy and transfer learning [52]. Along the same lines of reinforcement learning, transfer learning profits from a model already learned to apply the gained knowledge to new tasks. This technique has also been considered in the a-posteriori correction of models such as manifold learning in reduce order modeling of fluids [40] and dynamical optimization [35] [21] [24].

The success of the method proposed is conditioned by the characteristics of the dynamics learned in the source model. If the features are sufficiently general, and the algorithm has correctly captured the patterns of the fluid dynamics, the reinforcement learning scheme will smoothly transition to the new perceived liquid. Therefore, this method relies on the capacity of GENERIC as a pattern emergent algorithm to learn the major insights in data, regardless of the amount or the level of description of the data provided.

## 2 Problem formulation

We describe the evolution of a complex physical system as a function of a set of phase variables $s_i$ that define its state. The source model and correction algorithms are built employing the GENERIC equation [23]. It proposes a mesoscopic formulation of dynamical systems to describe their behavior in terms of the evolution of energy and entropy in the system. This approximation for dynamical systems can be discretized in time and its constituting terms can be thus inferred from data [19]. The so-called Structure-Preserving Neural Networks (SPNN) [27] [26] are deep learning architectures that employ the GENERIC formalism as an inductive bias to learn dynamical patterns from data and perform the simulations with guaranteed physically meaningful results.

Our method constructs in a first step a learned simulator, by employing a neural network trained offline with synthetic data coming from computational simulations [41]. This level of description agrees with theories that have demonstrated that human reasoning about fluid physics can be understood as a learned simulator operating at a coarse-grained level of description [6]. To train this first simulator, we employ data coming from full-field and high-fidelity
smooth particle hydrodynamics discretization of a base fluid under different initial conditions. We consider a constant fluid volume and, despite the scope of the correction, we only learn new behaviors, not new geometries, to bound the problem in this piece of work. Thus the initial discretization into particles is always the same. This assumption is employed to justify the neural networks used in the formulation. Although we consider a non-extremely chaotic behavior of the particles in the dynamical evolution, the selected fluids present rather different variations in the history of the free surface, enabling the correction of the models to adapt to the new observed physics.

This initial simulator is first trained off-line with synthetic data coming from the simulation of glycerine sloshing phenomena (viscosity \( \mu = 0.950 Ns/m^2 \) and density \( \rho = 1261 kg/m^3 \)) [22]. This learned simulator unveils the intrinsic dimensionality of data using a sparse autoencoder, to develop a reduced-order model, able to run under stringent real-time constraints [20]. Fig.1 describes the architecture of the original simulator. In this way, we compress the information within the initial dataset of full-field state vectors, which consist of the position, velocity, energy, and stress tensor of all the particles of the SPH discretization. This knowledge is transferred to two new networks. Then we train, on one hand, the time integrator of the evolution of the dynamics in the latent manifold based on structure-preserving NN.

Our learned simulator is then faced to on-line measurements performed by a commodity camera. During this online phase, we have access to the position of the free surface of the fluid only. We therefore substitute the encoder by a recurrent neural network to connect these partial measurements of the free surface of the liquid to the already unveiled latent manifold of the dynamics. We assemble these three networks to develop the simulation-in-the-loop system that, coupled with computer vision elements, will correct the results from new data acquired from observations.

2.1 Data acquisition

2.1.1 Computational datasets

We first conduct a set of purely computational experiments. This allows us to obtain accurate error measurements against high-fidelity simulations that are considered as the ground truth. Synthetic data are obtained from computational simulations using the software Abaqus (Dassault Systèmes, Simulia Corp).

The method has been tested against a total of ten different liquids, presenting both Newtonian and non-Newtonian behavior. We first train a learned simulator for glycerine [22]. With this simulator, we try to understand the behavior of water and melted butter, for instance, both described as Newtonian fluids. Although blood is sometimes considered to be Newtonian, it has been described as non-Newtonian [50]. It is considered to be shear thinning, becoming less viscous under the stress applied. The general description of the rheology and change of properties of fluids is performed based on the Herschel-Bulkley model [28]:

\[
\tau(t) = k\gamma^n(t) + \tau_0
\]

Where \( \tau \) is the shear stress, \( \tau_0 \) the yield stress, \( \gamma \) the shear rate, \( k \) the consistency index and \( n \) the flow index. For shear thinning fluids, \( \tau_0 = 0 \), and the flow index is \( n > 1 \). In the proposed case of computational blood, the computational liquid is defined by the constants \( k = 0.017 Pas, n = 0.708 \), and \( \tau_0 = 0 \).

These datasets are publicly available at https://github.com/beatrizmoya/sloshingfluids/.

2.1.2 Real world dataset

We then employ a stereo camera for data acquisition of the free surface of real-world liquids. The model chosen is a Real-Sense D435 https://www.intelrealsense.com/depth-camera-d435/. In a camera model, the correlation of the pixels in a picture, or frame, with the point they represent in the real world \( p_w \) is given by the extrinsic and the intrinsic parameters. The extrinsic parameters, rotation \( R \) and translation \( t \) matrices define the position and orientation of the camera referring to the world frame. It outputs the relative position of the camera in the real world. On the other hand, the intrinsic parameters \( K \) map the pixel coordinates to the camera coordinates. The camera provides the intrinsic and extrinsic parameters to convert the pixel coordinates on the image directly to three-dimensional coordinates they represent in the real world,

\[
\begin{bmatrix}
  u \\
  v \\
  1
\end{bmatrix} = \begin{bmatrix}
  f_x & 0 & c_x \\
  0 & f_y & c_y \\
  0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
  r_{11} & r_{12} & r_{13} & t_1 \\
  r_{21} & r_{22} & r_{23} & t_2 \\
  r_{31} & r_{32} & r_{33} & t_3
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix},
\]

or, in brief,

\[
\tilde{x}_s = K[R|t]p_w.
\]

The camera has stereoscopic depth technology to perceive the depth field of every pixel in the frame. With two lenses, triangulation [25] gives the 3D position of the points. Despite the accuracy of the camera, fine-tuning was necessary due to the lack of texture of the glass and liquids. These conditions difficult the tracking of points, resulting in errors. The application of filters (hole-filling and edge-preserving filters) improves the performance of depth estimation. In optimal conditions, the error in depth reconstruction is lower than 2% in a \( 0 \to 2m \) range. After the fine-tuning of the camera, in the threshold where we locate the glass for the recording, the camera noise in depth estimation is of the order of two millimeters.
Given a sufficient quality of depth resolution, we perform the observation and tracking of the free surface. We extract the profile of the free surface from binary (black and white) images. Under appropriate adjustment, this representation shows a black to white gradient between the liquid and the environment, see Fig. 2. These color changes enable to track the free surface. We extract the depth maps of the pixels of the aforementioned boundary. No further smoothing of the data is performed. Realtime tracking is performed with a camera resolution of 480 × 640 pixels.

From the acquired map, we project the pixel coordinates of the free surface to real-world coordinates and select their horizontal and vertical coordinates, or 2D real coordinates. Given the density of points detected on the free surface, we perform a simple linear interpolation to output the vertical displacement at equally spaced points of the free surface. This state information of the free surface is assembled in sequences to feed the source algorithm.

The dataset of the free-surface measurements performed on real liquids has been made publicly available: https://github.com/beatrizmoya/RLfluidperception

2.2 Physics-constrained learning

Once data are captured, either in computational, synthetic form or as a recording of the physical reality, our aim is to identify the state of the observed scene (perception) in terms of observable and non-observable variables—such as stresses, for instance—and to predict future states of the system (reasoning) by learned simulation. In the last years, a growing interest has been noticed in the incorporation of known physics in the form of inductive biases to this learning procedure. These representations will be essential for the generalization and interpretability of the method. Of particular importance, when some form of conservation (related to symmetries through Noether’s theorem) is present, this can be imposed to the learning procedure by invok-
ing Lagrangian or Hamiltonian frameworks. Despite their success, many dynamical systems are also compromised by dissipative effects, which implies that this framework of techniques is no longer valid for application. This premise also matches the dissipative nature of the real world and the entropy that emerges from the lack of information. Fluid dynamics, as well as other dissipative dynamical systems, own a so-called metriplectic structure [33]. This formalism is suitable for cases in which the conservative Hamiltonian description of a dynamical system includes unresolved degrees of freedom, that are not included in mesoscopic descriptions and that introduce dissipation by the fluctuation-dissipation theorem [34]. Consider an initial microscale description at the molecular dynamics scale, that can be expressed in terms of a purely conservative formulation. The degrees of freedom, and therefore knowledge, that we omit growing from the micro to the meso and macro scales introduce dissipation that is included in the metric part of the formulation.

This structure-preserving formulation guarantees the conservation of critical quantities (mass, momentum), and the thermodynamical admissibility of the evolution of the system under study. GENERIC is the chosen thermodynamical framework to derive a structure-preserving time integrator for describing the time evolution of a system concerning the evolution of its energy and entropy functionals [23]. As a consequence, this framework is also valid for dynamical systems that go beyond equilibrium in a thermodynamical context. The GENERIC formulation is usually written as follows:

\[
\frac{ds}{dt} = L \nabla E + M \nabla S.
\]

Here, \(s\) denotes a set of independent state variables that fully describe the thermodynamical state of the fluid. Without that information, we lack a GENERIC structure. Fluid dynamics are fully described in terms of the position and momentum of the particle discretization, internal energy, and, in the case of learning more complex fluids, the extra-stress tensor related to their microscopic evolution [15]. \(E(s)\) and \(S(s)\) are the global energy and entropy of the system. \(L(s)\) is the Poisson matrix. It is antisymmetric, and together with the gradient of energy \(\nabla E\), characterizes the reversible part of the studied dynamics. \(M(s)\) is the friction matrix, which describes the dissipative irreversible characteristics of the system in conjunction with the entropy gradient \(\nabla S\). \(M\) is symmetric positive semidefinite.

On top of the aforementioned formulation and matrix descriptions, we must also ensure the fulfillment of the so-called degeneracy conditions:

\[
L \frac{\partial S}{\partial s} = M \frac{\partial E}{\partial s} = 0,
\]

that guarantee that the energy \(E\) is not involved in the entropy production of the dissipative part of the dynamics, and that the entropy \(S\) does not contribute to the energy conservation.

The GENERIC equation can be discretized in time by a forward Euler scheme formulated in time increments \(\Delta t\), and the resulting formulation subsequently inferred from data:

\[
s_{n+1} = s_n + \Delta t \left( L_n \frac{DE}{Ds_n} + M_n \frac{DS_n}{Ds_n} \right),
\]

subjected to the degeneracy conditions:

\[
L_n \cdot \frac{DS_n}{Dx_n} = 0, \quad M_n \cdot \frac{DE_n}{Dx_n} = 0,
\]

ensuring the thermodynamical consistency of the resulting model.

Structure-preserving neural networks (SPNN) embed GENERIC into a deep learning architecture to unveil the value of the main elements of the equation [26] [27]. In this framework, the discretized gradients of energy and entropy are the targets of the optimization of the net. In addition, although \(L\) and \(M\) are usually known in the literature, they are unknown in the low dimensional manifold
where we simulate the dynamics. Hence, the network learns \( L, M, \frac{DE_n}{DS_n}, \frac{DS_n}{DS_n} \) from data. This is represented in Fig. 1. If the network is considered the agent of the RL algorithm, the actions are the GENERIC elements that the network proposes given a current dynamical state to perform the integration of the dynamics in time.

### 2.3 Reinforcement learning algorithm

As already stated, our system is initially trained to perceive and reason about glycerine. But it is then faced to different, previously unseen fluids. To be able to provide with useful and credible predictions for these new fluids, a reinforcement learning strategy is employed. To proceed with it, we have to accomplish two tasks: projecting the problem to the reinforcement learning framework, and then designing the strategy to profit from it. Reinforcement learning is the theory that optimizes the correlation between actions and the state of an environment from the interaction. The elements of reinforcement learning are itemized and related to the sloshing dynamics problem below.

![Figure 3: Reinforcement learning scheme. Given the partial observation of the free surface \( z_n \) at time \( t = n \times \Delta t \) our method estimates the full state of the fluid \( \hat{s}_{n+1} \). Its similarity with the next state \( s_{n+1} \) is evaluated to update the model. To preserve the patterns previously learnt, only a few layers are activated (in yellow) for backpropagation of the error and correction of the model.](image)

The environment, or sloshing dynamics, is described by its dynamical state \( s_n \in S \), where \( S \) represents the state space where the dynamics are embedded. From that state \( s_n \), we actually have access to a partial observation \( z_n \in Z \). If \( s_n \) stands for the fully dynamical description of the slosh, \( z_n \) stands for the observation of the free surface. Free surface detection and tracking are performed employing computer vision techniques. The environment then transitions to the next state \( s_{n+1} \). The agent is the neural network itself. The new state is reached based on the action \( a \), which consists in the computation of the GENERIC parameters by the neural network. The ultimate objective to accomplish in this work is to find the most convenient policy \( \pi(a|s) \) so that we obtain a satisfactory result of the motion reconstruction of the free surface.

The behavior policy \( \pi_\theta(a|s) \) is the relationship between the learned simulator proposed by the network and the results of the model with regard to the parameters \( \theta \) of the neural network. Specifically, the action proposed is the selection of \( L, M, \frac{DE_n}{DS_n}, \frac{DS_n}{DS_n} \) given a state \( s_n \).

The reward is related to the accuracy of the free surface motion reconstruction. Through this, we reward the improvements of the network that lead to a closer result to the ground truth. We choose an \( L_2 \) norm for this purpose. We also reward those solutions that fulfill the degeneracy conditions established in the GENERIC formalism. As a result, the reward function is composed of these two terms weighted to control the influence of each in the optimization of the policy. The degeneracy condition is thus considered as a soft constraint:

\[
  r_\pi = \lambda \frac{1}{N} \sum_N \left( \| z_{n+1} - \hat{z}_{n+1} \|^2 + \frac{1}{N} \sum_N \left( \| L_n \frac{DE_n}{DS_n} \|^2 + \| M_n \frac{DS_n}{DS_n} \|^2 \right) \right).
\]  

At each loop of the training, the algorithm is fed with \( N \) data samples coming from the interaction with the environment. With them, the behavior policy parameters are updated. This process is repeated until we reach convergence in the adaptation process.

#### Algorithm 1 General Reinforcement learning pseudocode

Require: Free surface information \( z_n \in Z \) as a subset of the full dynamical state \( s_n \in S \), and the source model as initial behavior policy \( \pi_\theta \)

Ensure: Next dynamical states in time \( s_{n+1} \)

for iterations until convergence do
  for \( n=1 \) to \( N \) sequences do
    Encoder \( \mathcal{E} \leftarrow \phi(z_n) \),
    Compute forward propagation and determine action \( a_n \leftarrow \pi_\theta(\mathcal{E}_n) \), being \( a_n \) the tuple \( a_n = [L_n, M_n, \frac{DE_n}{DS_n}] \),
    Determine next integration step \( \hat{\mathcal{E}}_{n+1} \leftarrow \Delta t(L_n \frac{DE_n}{DS_n} + M_n \frac{DS_n}{DS_n}) + \mathcal{E}_n \),
    Decoder \( \hat{s}_{n+1} \leftarrow \psi(\hat{\mathcal{E}}_{n+1}) \),
  end for
  Compute reward \( r_\pi \),
  Backward propagation and update RL policy \( \pi_\theta \),
end for
return RL optimized policy \( \pi_\theta \)

As described in the algorithm, we first collect information coming from the observation of a fluid. Then, we com-
pute the actions to obtain $L, M$ and perform the time integration for each given state. We then compute the reward with this information, and perform back propagation to update the behavior policy of the neural network.

The backpropagation is computed through selected layers of the whole source model. This approach, in conjunction with low learning rates, ensure the minimum loss of information from the previous model to preserve the insights already learned with GENERIC. We profit from these known patterns to deal with the limitations coming from partial observations and low data regimes.

3 Initial learned simulator

The base simulator for glycerine is trained with synthetic data coming from the simulation of a fluid discretized in particles based on the smooth particle hydrodynamics technique. This coarse level of description is appropriate to have sufficient information about the dynamics to train learned simulators without compromising time. The state of the fluid is described by a set of state variables (position, velocity, energy, and stress tensor of each particle), evaluated at each particle of the discretization.

Due to the complexity of the problem, the simulator involves three different steps, and transfer learning is applied to carry the knowledge from simpler architectures to more elaborated ones. First, we train a fully-connected autoencoder to embed the data in a low dimensional manifold where we will perform the integration in time. It consists of two parts: the encoder and the decoder. The encoder learns a mapping $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^d$ to a low dimensional manifold where the dynamics are embedded. The decoder $\psi : \mathbb{R}^d \rightarrow \mathbb{R}^D$ has the same structure of the encoder, but inverted, to project state vectors of the latent space to the full order space. Not only the computational cost for training the integrator will be reduced, but also the pre-processing will facilitate the learning process. By applying this model order reduction we are already triggering the emergence of the patterns, and encouraging the net to learn the main features of data. We train five autoencoders in parallel, one per each group of the already mentioned state variables, to capture the intrinsic insight of each one. The final latent manifold results from merging the latent subspaces of each autoencoder. The bottleneck of each autoencoder is truncated to reduce the dimensionality by forcing sparsity on its learning. Thus, for a given bottleneck, only a few latent values have a resemblant order of magnitude greater than zero. The decoder mirrors the encoder. The resulting latent manifold has 13 dimensions. The decoder is part of the new model, but it is not considered in the RL optimization. Sparsity is imposed with a $L_1$ regularizer that penalizes activation in the bottleneck to enforce a higher reduction. This penalization complements the mean squared error reconstruction loss of the solution of the autoencoder:

$$L_{AE} = \frac{1}{N} \sum_{i=1}^{N} (z_i - \hat{z}_i)^2 + \lambda \sum_{i=1}^{N} |x_i|.$$ 

The sparsity loss is weighted by a lambda factor to control its influence on the learning process. In the source model, the weight factor is defined within the range $\lambda = 0.001 - 0.005$. As we train an autoencoder for each group of state variables, it is chosen according to each case.

Since the full-field state of the fluid cannot be evaluated, i.e., we cannot measure with a camera the internal energy of points of the fluid, we train a mapping from the available information of the free surface to the latent manifold in a second step of the procedure. Hence, we transfer the information from the latent manifold previously obtained to a recurrent neural network to substitute the encoder. The recurrent neural network exploits the analysis of sequences of data to distill the insights about the dynamics that enable to find a mapping between the observations and the latent manifold, which contains the full information about the dynamics embedded. We employ a gated recurrent unit (GRU) structure [11]. We need a sequence of at least 16 snapshots to relate the measurements of the free surface with the latent manifold. The net consists of three GRU hidden layers of 26 neurons, and one last feed-forward fully connected layer to connect the last GRU layer to the latent space of size 13. The training is guided by the mean squared error evaluation of the accuracy in the reconstruction of the latent vector.

Finally, the structure-preserving neural network is trained for time integration of the dynamics based on the reconstruction accuracy of the solution in the latent manifold, as well as the compliance of the degeneracy conditions. This network is composed of 13 hidden layers of size 195 with ReLU activations but for the first and last layers, which have linear activations. The loss used for the training includes the mean squared error reconstruction error of the prediction and the loss related to the degeneracy conditions. The reconstruction error is weighted by a $\lambda$ factor to prioritize these terms. In this model, $\lambda_{spnn} = 10^3$.

4 Results

4.1 Reinforcement learning in a purely computational scenario

The employ of pseudo-experimental data coming from simulations allows us to compute precise error measurements on the different adaptation strategies and their effect on diverse quantities of interest. Given a learned simulator trained off-line for glycerine data, we first apply the RL methodology to learn three different liquids presenting
rather diverse behaviors: water and butter as Newtonian fluids, and blood, simulated as a non-Newtonian liquid. We employ synthetic data consisting in snapshots taken from four simulations for each liquid performed under different velocity conditions to trigger diverse sloshing results. From this information, we prepare the dataset with sequences of the positions of the particles that belong to the free surface. I.e, we track the free-surface particles, take their position, and prepare sequences of 16 snapshots for each time step $\Delta t$ (the time instant of interest and the 15 previous states of the free surface). The length of 16 was found to be the minimum length of the sequences to find an embedding of the free surface measurements on each fluid’s latent manifold, already computed off-line. The dataset of water has 750 snapshots in total, and butter and blood, 480. Synthetic data is available at a sampling frequency of $200 \text{Hz}$, which stands for a time step of $\Delta t = 0.005$ seconds. Therefore, the source model was built for this availability of data and time step. Nevertheless, the snapshots are sampled at every frequency of $60 \text{Hz}$, or equivalently $\Delta t = 0.015$ seconds, that matches the performance frequency of the commodity camera that will be employed in the real scenario. Hence, the change in the time step has also an effect on the model that should be corrected. The sequences of the dataset are randomly split into two subsets: 80% for training and 20% for testing.

We carry out the RL correction in the integration scheme and the embedding onto a lower-dimensional manifold. The correction is accomplished by activating the backpropagation in the last layer of the GRU network, and the 4 last layers, out of 13, of the SPNN. Since those are precedent layers of the decoder, and we train the network as a whole, we achieve the desired reconstruction with no need of altering the last structure of the network. The adaptation converges after 2000 epochs, at a small learning rate $lr = 0.0005$ and weight decay $wd = 0.00001$. We choose Adam optimizer [32]. The reconstruction is weighted by a factor $\lambda = 2000$. Fig.4 shows the transition from the original manifold, trained off-line, to the new latent space that fits the emulated dynamics. Since the liquids bear a resemblance with the original liquid, glycerine, their manifolds do not show drastic changes in their structure compared to the initial solution. This fact also highlights the generality of the patterns of the dynamics learned in the source glycerine simulator.

The height reference is placed at the bottom of the glass and, therefore, the relative error is computed with the total height of the points of the free surface selected for the evaluation:

$$\text{error} = \frac{1}{N} \sqrt{\sum_{n=1}^{N} \frac{z_n^2 - \hat{z}_n^2}{z_n^2}},$$

with $N$ the number of samples in the dataset, and $z_n$ and $\hat{z}_n$ the ground truth and simulated free surface data, respectively.

The relative error is represented in Fig.5. The RL algorithm improves the performance of the simulator, reducing significantly the reconstruction error of the free surface. The relative error is also presented for the complete dataset of each liquid, reconstructing the 80% employed in the correction, and the 20% considered for test purposes. The four simulations can be distinguished in the graph. The error peaks appear in the correspondent top peaks of the sloshing of each simulation, where the surface deformation is greater. Nevertheless, the error continues to be in low ranges. Water presents higher sloshing than glycerine. Thus, the desired solutions were out of the source database. In this
In the case, the maximum error drops from up to 12% to 4−6%, remaining at an average of not more than 3%. Butter reconstruction error is reduced from 6% to less than 2%. Finally, the algorithm manages accurately to evolve from a Newtonian to a non-Newtonian fluid to replicate the behavior of blood. For this liquid, the maximum relative error goes from 7% to 3.5%. It is worth mentioning that, providing that we employ a meshless method (SPH) to generate synthetic data, we can observe some errors coming from the different particle distributions. Although the movement matches the ground truth, we can have two different particle configurations. This results in different interpolated free surfaces that cause little errors despite their strong resemblance.

For sloshing control problems, the maximum height of the free surface is the control variable to reward the actions that ensure the closest to sloshing-free solutions. For this reason, we also analyze the performance of the correction by evaluating the maximum height of the slosh in the most critical simulation for each liquid in Fig. 6. This detail is correlated with the correspondent segment of the relative error graphs. Finally, it is worth highlighting that the reconstruction not only adjusts better to the magnitude of the slosh but also their occurrence in time. Before the correction, some peaks appear delayed or are skipped.

### 4.2 Adaptation in an real scenario

Employing pseudo-experimental data coming from simulations has the great advantage of allowing the computation of precise error measurements. However, the ultimate objective of this work is to develop a methodology for real-life scenarios. In this scenario, the algorithm must adapt to previously unseen real liquids with different properties that result in different frequencies and magnitudes of slosh and dissipation and duration of slosh. This correction will be done from the evaluation of the free surface of liquids tracked utilizing computer vision. This problem showcases various challenges such as errors in depth estimation, the difficulty to detect and track the free surface, the complexity of real liquids to capture all the dynamical features from the recorded videos, and user actuation. The last statement refers to the direction of the movement of the liquid. In this work, we limit the experiment to a plane movement in two possible directions (move the glass to the left or the right), but we can experience slight deviations in the actuation. We impose this restriction to bound the problem to a case where we can evaluate properly the free surface from a fixed position of the camera with the detection method employed.

Four different liquids are evaluated in the reinforcement learning framework here proposed: water, honey, beer, and gazpacho (traditional Spanish cold soup). These four materials cover a wide spectrum of behaviors and textures, from lower to higher viscosities than glycerine. In addition, the liquids selected are daily products of interest in real manipulation, that can show different behaviors depending on the production process. The glass employed is of the same shape as the computational container employed in the simulations of the source model, and the vessel is filled approx-
Figure 6: Fluid reconstruction before and after RL correction of water blood and butter. Representation of the sloshing peaks of the ground truth (in blue) compared to the network’s results before (in red) and after (in yellow) applying reinforcement learning to adapt from observations in selected critical snapshots, indicated in the figure.

There are two recordings for each selected liquid. The first video, divided into 80% snapshots for training and 20% for validation to prevent overfitting, is employed in the RL algorithm. The second video evaluates the performance of the new models over unknown datasets. Given these data, we apply the reinforcement learning scheme for each liquid individually, obtaining four new simulators, adapted for each scenario. Equivalently to the computational test case, the correction is performed in selected active layers. Considering the complexity of the liquids’ behavior and the noise of the samples, the last layer of the recurrent neural network (GRU) and the 5 last layers, out of 13, of the SPNN are activated for the correction. The correction of the recurrent neural network will adapt the manifold to the new liquid, while the partial activation of the SPNN will adapt the simulation to the new dynamics observed. The real benchmarks require more correction than the computational benchmarks. However, we maintain low learning rates and partial activation to prevent the network from learning the measurements’ noise and forgetting the previously learned dynamical patterns. The correction converges after 4000 epochs, at a small learning rate $lr = 0.0001$ and weight decay $wd = 0.00001$. The reconstruction is weighted by a factor $\lambda = 2000$.

Fig. 7 showcases the results of the correction over training and testing recordings of the four liquids. The four exhibit an improvement of the sloshing reconstruction compared to the performance of the source model before correction. In addition, the temporal integration with data from the test datasets presents a noteworthy performance considering that this information is new to the network. The training recordings are shorter than 10 seconds, and only three or four sloshes are captured in these datasets. However, the new model learns the new target behavior. Even though we work in a low-data regime to perform the correction of complex liquids, the method successfully reproduces the train and test benchmarks. These results are obtained because of the inductive biases learned and preserved in the correction, leading to an efficient correction from limited data and partial measurements, in this case only considering the free surface for the reconstruction.

Water and gazpacho present slightly higher errors than the other liquids in the test dataset. This is probably due to the difficulties experienced in the data acquisition and noise of the samples in the experiment. Additionally, water has higher slosh frequencies and longer slosh time. Therefore, more varied sloshes would be needed to reduce the error, still in a low-data regime. Despite this, the method already performs the adaptation over train and test information correctly.

Finally, Fig. 8 shows renders of the simulation results before and after correction compared to the snapshot that they predict in time. Additionally, the renders are complemented by a comparison between the particle discretized solution and the ground truth’s free surface. The absolute error is indicated in mm in a colored scale to analyze visually the magnitude of the distortions more accurately. It is worth mentioning that, although the particle configuration could be distorted after training to match the free surface (since
Figure 7: Results of the adaptation to new computational liquid for water beer gazpacho and honey. The relative errors of the reconstruction are presented for the training database and over the information of an unseen recording. Despite the complexity and the few data employed, the correction in the test dataset is also noticeable. Relative error representations are accompanied by the maximum height reconstruction of one of the sloshes perceived to compare the performance of before and after correction.

5 Conclusion

The present work shows a reinforcement learning methodology guided by GENERIC as an inductive bias for perception and reasoning about fluid sloshing. The algorithm learns from observations to accurately mimic new fluid behaviors from the sole observation of the free surface. The method provides a tool for model inference with real data from partial observations of complex dynamics. The correction of physics perception enables the machine to adapt and learn previously unseen liquids present in daily tasks with unknown properties. We start from a source model trained with computational data to learn a physically sound simulator of the sloshing dynamics upon GENERIC to ensure the physical consistency and generalization of the results. The calculation is performed in a low-dimensional manifold to ensure real-time performance. Thus, we obtain real-time interaction with the environment in which the

it is the only information available) losing the sense of the geometry, the method manages to preserve the fluid volume configuration. The time is indicated to correlate each render to the error in Fig. 8. Despite the correction, the algorithm outputs a matching shape with the real entity. As a result, giving only the free surface of the observed fluids, the networks integrate in time the dynamical evolution of the liquid and provide a three-dimensional reconstruction of the fluid volume. The particle discretization represents a fluid volume that can be translated to the user through renderization and augmented reality. This representation bridges the gap between reality and the virtual environment to provide augmented information to the machine, and the user, for decision making.
Figure 8: Render of the volume reconstruction before and after correction. The algorithm outputs a particle discretization of the fluid that can be presented as a three-dimensional render of the volume for visualization and interpretation. The figure shows the peaks of the dynamics observed in a piece of the recordings.
model, or digital twin of the real liquid, operates to have response capacity.

We illustrate the benefits of physics-informed deep learning for reinforcement learning and correction. Given an off-line learned simulator for one particular fluid (glycerine, in our case), our method manages to evolve to a new representation of the dynamics to match the behavior of previously unseen liquids. The recordings employed are limited to 10 seconds approximately. In this low-data regime, the method adapts to the new dynamics perceived. The good performance of the method can be observed also in the test recording, which has not been seen by the network before. The success is attributed to the insights learned in the source model with simulation data, and the inductive bias imposed by GENERIC, that ensures the fulfillment of the principles of thermodynamics. These are sufficiently general and precise to allow the simulator to evolve smoothly to new liquids.

A challenge in physics perception is the balance between adaptivity and the risk of learning noise coming from the experimental nature of the data acquisition technique. Liquids and vessels are non-Lambertian, i.e., they do not have a diffusely reflecting surface, or matter, convenient for depth estimation. Despite the fine-tuning of the camera, the measurements include noise and invalid measurements from which the free surface has to be reconstructed. By applying transfer learning and performing slow training we have prevented the network from learning meaningless information coming from measurements. Moreover, the patterns already learned help reconstruct the information to learn the new behaviors accurately.

Despite the accuracy observed in the reconstruction of train and test recordings, the performance of the method could be further improved over the last model learned by retraining with new datasets acquired by the same means. Hence, it can perform corrections to not only persist in the improvement of the reconstruction but also to adapt to the evolving nature of the scenario.

The results observed can be a starting point to adapt the method to new geometries and include this new parameter in the optimization from the application of geometric deep learning. In addition, this problem could be further extended with more general liquid detection techniques [48] [13] capable of analyzing transparent and not textured elements from different perspectives.

6 Citations

References

[1] K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Rapid trial-and-error learning with simulation supports flexible tool use and physical reasoning. *Proceedings of the National Academy of Sciences*, 117(47):29302–29310, 2020.

[2] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Pfaff, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[3] C. G. Atkeson and S. Schaal. Learning tasks from a single demonstration. In *Proceedings of International Conference on Robotics and Automation*, volume 2, pages 1706–1712. IEEE, 1997.

[4] J. Ayensa-Jíménez, M. H. Doweidar, J. A. Sanz-Herrera, and M. Doñalé. Prediction and identification of physical systems by means of physically-guided neural networks with meaningful internal layers. *Computer Methods in Applied Mechanics and Engineering*, 381:113816, 2021.

[5] M. Bassenne and A. Lozano-Durán. Computational model discovery with reinforcement learning. *arXiv preprint arXiv:2001.00008*, 2019.

[6] C. J. Bates, I. Yildirim, J. B. Tenenbaum, and P. Battaglia. Modeling human intuitions about liquid flow with particle-based simulation. *PLoS computational biology*, 15(7):e1007210, 2019.

[7] P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

[8] M. Benosman, A. Chakrabarty, and J. Borggaard. Reinforcement learning-based model reduction for partial differential equations: Application to the burgers equation. In *Handbook of Reinforcement Learning and Control*, pages 293–317. Springer, 2021.

[9] T. Bertalan, F. Dietrich, I. Mezić, and I. G. Kevrekidis. On learning hamiltonian systems from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(12):121107, 2019.

[10] M. A. Bucci, O. Semeraro, A. Allauzen, G. Wsiewowski, L. Cordier, and L. Mathelin. Control of chaotic systems by deep reinforcement learning. *Proceedings of the Royal Society A*, 475(2231):20190351, 2019.

[11] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
[12] M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–423, 2013.

[13] C. Do, T. Schubert, and W. Burgard. A probabilistic approach to liquid level detection in cups using an rgb-d camera. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2075–2080. IEEE, 2016.

[14] J. Eichelsdörfer, S. Kaltenbach, and P.-S. Koutsourakis. Physics-enhanced neural networks in the small data regime. *arXiv preprint arXiv:2111.10329*, 2021.

[15] P. Espanol, M. Serrano, and H. C. Öttinger. Thermodynamically admissible form for discrete hydrodynamics. *Physical review letters*, 83(22):4542, 1999.

[16] P. Garnier, J. Viquerat, J. Rabault, A. Larcher, A. Kuhnle, and E. Hachem. A review on deep reinforcement learning for fluid mechanics. *Computers & Fluids*, 225:104973, 2021.

[17] B. Goertzel and C. Pennachin. *Artificial general intelligence*, volume 2. Springer, 2007.

[18] D. González, F. Chinesta, and E. Cueto. Learning corrections for hyperelastic models from data. *Frontiers in Materials*, 6:14, 2019.

[19] D. González, F. Chinesta, and E. Cueto. Thermodynamically consistent data-driven computational mechanics. *Continuum Mechanics and Thermodynamics*, 31(1):239–253, 2019.

[20] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[21] S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. *Theoretical and Applied Fracture Mechanics*, 106:102447, 2020.

[22] S. Gregory. Physical properties of glycerine. In *Glycerine*, pages 113–156. CRC Press, 2018.

[23] M. Grmela and H. C. Öttinger. Dynamics and thermodynamics of complex fluids. i. development of a general formalism. *Physical Review E*, 56(6):6620, 1997.

[24] L. Guastoni, A. Güemes, A. Ianiro, S. Discetti, P. Schlatter, H. Azizpour, and R. Vinueza. Convolutional-network models to predict wall-bounded turbulence from wall quantities. *Journal of Fluid Mechanics*, 928, 2021.

[25] R. Hartley and A. Zisserman. Multiple view geometry in computer vision. *Cambridge university press*, 2003.

[26] Q. Hernandez, A. Badias, D. Gonzalez, F. Chinesta, and E. Cueto. Deep learning of thermodynamics-aware reduced-order models from data. *Computer Methods in Applied Mechanics and Engineering*, 379:113763, 2021.

[27] Q. Hernández, A. Badías, D. González, F. Chinesta, and E. Cueto. Structure-preserving neural networks. *Journal of Computational Physics*, 426:109950, 2021.

[28] W. H. Herschel and R. Bulkley. Konsistenzmessungen von gummi-benzollösungen. *Kolloid-Zeitschrift*, 39(4):291–300, 1926.

[29] T. Hester and P. Stone. Learning and using models. In *Reinforcement learning*, pages 111–141. Springer, 2012.

[30] S. Huang, Z. He, and C. Reina. Variational onsager neural networks (vonns): A thermodynamics-based variational learning strategy for non-equilibrium pdes. *arXiv preprint arXiv:2112.09085*, 2021.

[31] N. R. Ke, A. Singh, A. Touati, A. Goyal, Y. Bengio, D. Parikh, and D. Batra. Learning dynamics model in reinforcement learning by incorporating the long term future. *arXiv preprint arXiv:1903.01599*, 2019.

[32] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[33] M. Kraus. Metriplectic integrators for dissipative fluids. In *International Conference on Geometric Science of Information*, pages 292–301. Springer, 2021.

[34] R. Kubo. The fluctuation-dissipation theorem. *Reports on progress in physics*, 29(1):255, 1966.

[35] R. Laroche and M. Barlier. Transfer reinforcement learning with shared dynamics. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[36] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[37] C. K. Liu and D. Negrut. The role of physics-based simulators in robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:35–58, 2021.

[38] X.-Y. Liu and J.-X. Wang. Physics-informed dynastyle model-based deep reinforcement learning for dynamic control. *Proceedings of the Royal Society A*, 477(2255):20210618, 2021.
[39] F. Masi, I. Stefanou, P. Vannucci, and V. Maffi-Berthier. Thermodynamics-based artificial neural networks for constitutive modeling. *Journal of the Mechanics and Physics of Solids*, 147:104277, 2021.

[40] M. Mohoubujumman, L. G. Rebholz, and T. Iliescu. Physically constrained data-driven correction for reduced-order modeling of fluid flows. *International Journal for Numerical Methods in Fluids*, 89(3):103–122, 2019.

[41] B. Moya, A. Badías, D. Gonzalez, F. Chinesta, and E. Cueto. Physics perception in sloshing scenes with guaranteed thermodynamic consistency. *arXiv preprint arXiv:2106.13301*, 2021.

[42] G. Novati, H. L. de Laroussilhe, and P. Koumoutsakos. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1):87–96, 2021.

[43] J. Oh, X. Guo, H. Lee, R. Lewis, and S. Singh. Action-conditional video prediction using deep networks in atari games. *arXiv preprint arXiv:1507.08750*, 2015.

[44] R. Parr, L. Li, G. Taylor, C. Painter-Wakefield, and M. L. Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 752–759, 2008.

[45] J. Rabault and A. Kuhnle. Accelerating deep reinforcement learning strategies of flow control through a multi-environment approach. *Physics of Fluids*, 31(9):094105, 2019.

[46] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[47] F. Ren, J. Rabault, and H. Tang. Applying deep reinforcement learning to active flow control in weakly turbulent conditions. *Physics of Fluids*, 33(3):037121, 2021.

[48] S. Sajjan, M. Moore, M. Pan, G. Nagaraja, J. Lee, A. Zeng, and S. Song. Clear grasp: 3d shape estimation of transparent objects for manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3634–3642. IEEE, 2020.

[49] C. Schenck and D. Fox. Perceiving and reasoning about liquids using fully convolutional networks. *The International Journal of Robotics Research*, 37(4-5):452–471, 2018.

[50] M. Smith. Approximate viscosities of some common liquids.

[51] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[52] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

[53] S. Verma, G. Novati, and P. Koumoutsakos. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proceedings of the National Academy of Sciences*, 115(23):5849–5854, 2018.

[54] T. Wang, X. Bao, I. Clavera, J. Hoang, Y. Wen, E. Langlois, S. Zhang, G. Zhang, P. Abbeel, and J. Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[55] Y. Wang, Z. Shen, Z. Long, and B. Dong. Learning to discretize: solving 1d scalar conservation laws via deep reinforcement learning. *arXiv preprint arXiv:1905.11079*, 2019.

[56] H. Yu, X. Tian, E. Weinan, and Q. Li. Onsagernet: Learning stable and interpretable dynamics using a generalized onsager principle. *Physical Review Fluids*, 6(11):114402, 2021.

[57] H. Zhang, H. Nguyen, X.-N. Bui, B. Pradhan, P. G. Asteris, R. Costache, and J. Aryal. A generalized artificial intelligence model for estimating the friction angle of clays in evaluating slope stability using a deep neural network and harris hawks optimization algorithm. *Engineering with Computers*, pages 1–14, 2021.

[58] Z. Zhang, Y. Shin, and G. E. Karniadakis. Gfinns: Generic formalism informed neural networks for deterministic and stochastic dynamical systems. *arXiv preprint arXiv:2109.00092*, 2021.

[59] Y. D. Zhong, B. Dey, and A. Chakraborty. Symplectic ode-net: Learning hamiltonian dynamics with control. *arXiv preprint arXiv:1909.12077*, 2019.