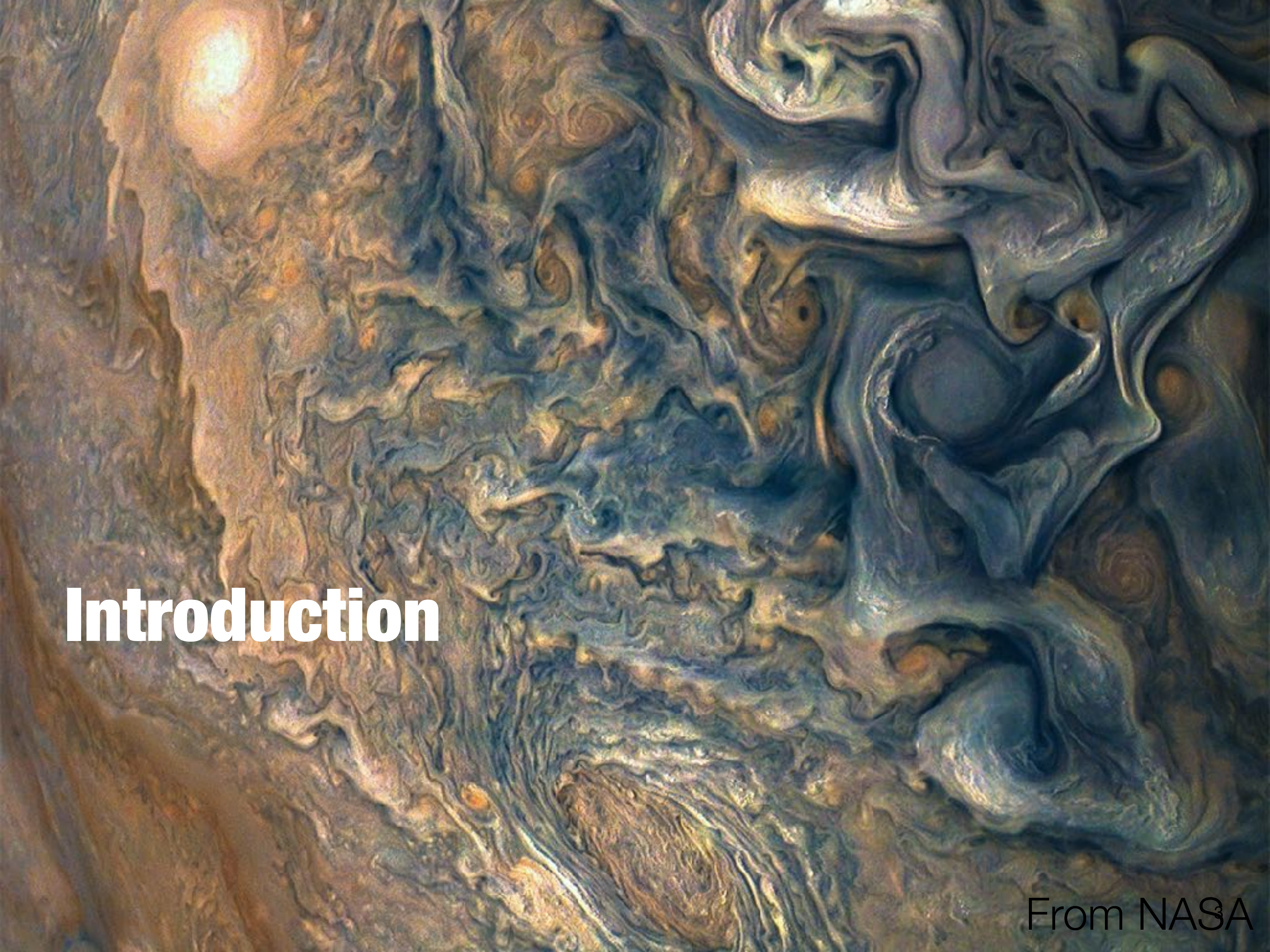
Deep Learning for Physical Processes: Incorporating Prior Scientific Knowledge

Application to forecasting Sea Surface Temperature

E. De Bézenac, A. Pajot, P. Gallinari

arthur.pajot@lip6.fr

Summary



Introduction

From NASA

Physical Model

- Large scientific background about natural phenomena modeling.
- Rely on differential equations, discretizations, and data assimilation, which are mature fields.
- Complex to develop.
- A lot of hyper-parameters have to be manually selected.
- Rely on prior knowledge of the natural process taking place.
- **Not an optimal exploitation of the data**

Deep Learning

- Prior agnostic approach
- Necessitates large amounts of data.
- Models are easy to develop and implement due to widely used Deep Learning platforms (Pytorch, Tensorflow,Keras).
- Implementation is efficient, making use of parallel computing (GPUs).
- Works very well for a wide range of tasks, state of the art in vision, translation, etc…
- Not competitive for natural complex phenomena modeling.
- **Not an optimal exploitation of background scientific knowledge.**

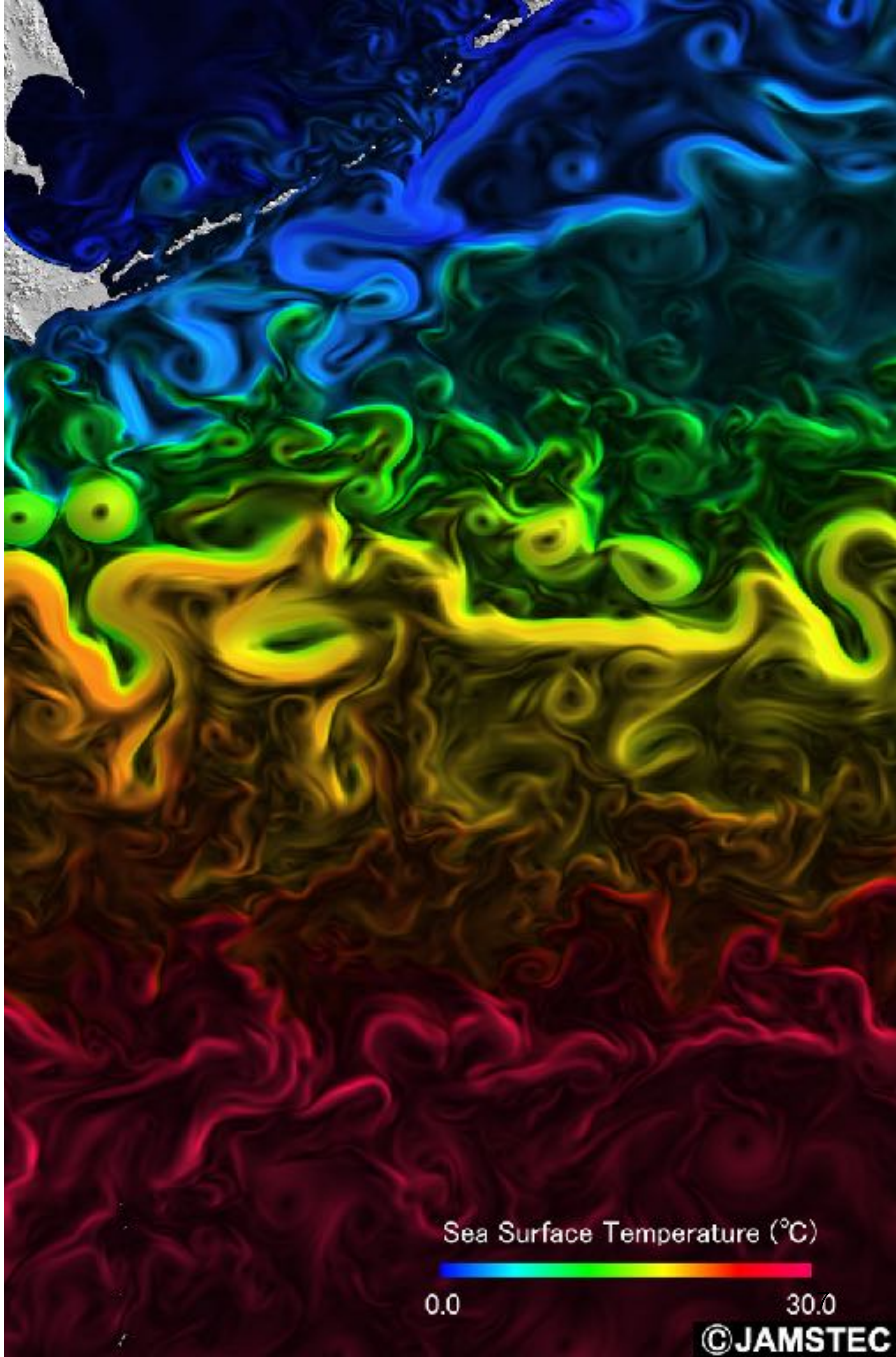
Emergence of hybrid models ?

Could we combine the physical and statistical paradigms to obtain the best of both worlds?

A new field of research seems to be appearing, merging ideas from Deep Learning and Physical models for analyzing natural data (e.g. Deep Learning for Physical Sciences Workshop at NIPS 2017, today…)

Case study : Sea Surface Temperature (SST) Forecasting

- Forecasting SST is critical in various applications, such as weather forecasting, of planning of coastal activities
- Large scale SST acquisitions are made possible with satellites.
- The problem is complex and high dimensional.



Model

Advection Diffusion Equation

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I$$

Advection Diffusion Equation

$$\frac{\partial I}{\partial t} + (w.\underset{\uparrow}{\nabla})I = D\nabla^2 I$$

Advective Term

Responsible for the transport of I by the bulk motion w.

Advection Diffusion Equation

$$\frac{\partial I}{\partial t} + \underbrace{(w.\nabla)I}_{\text{Advective Term}} = \underbrace{D\nabla^2 I}_{\text{Diffusive Term}}$$

Advective Term

Responsible for the transport of I by the bulk motion w.

Diffusive Term

Responsible for movement of particles from high concentration to regions with low concentration.

Advection Diffusion Equation

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I$$

Close form solution

$$I_t(x) = \int_\Omega k(\, x - w(x),\, y)\; I_0(y)\, dy$$

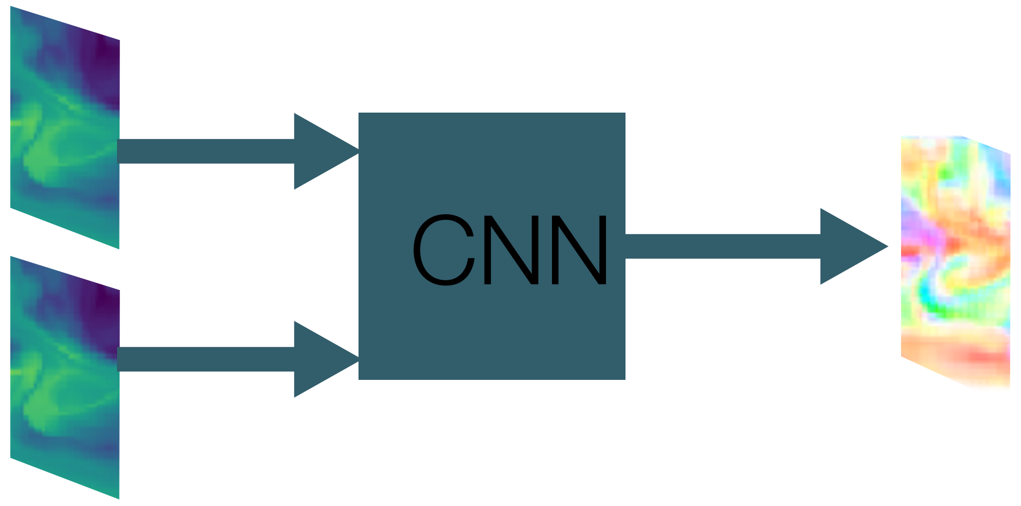
Where I_0 corresponds to the initial condition of I and k is an RBF kernel :

$$k(x - w, y) = \mathcal{N}(y \,|\, x - w,\ 2Dt)$$

I could be computed from I_0 and w but w is unknown

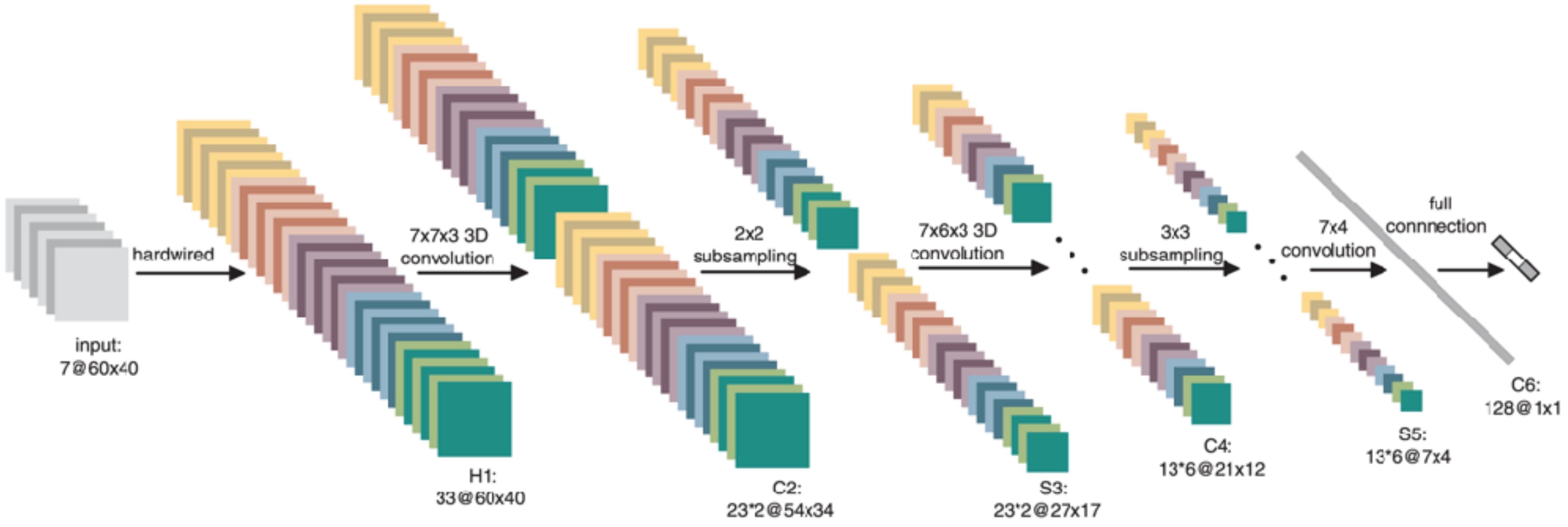
Motion estimation using a CNN

How do we estimate the motion field ?



- In data assimilation, a model on the motion field and a tracer equation on temperature is given and an energy functional is minimized.
- Here, we do not use any prior model for the motion field, but rather estimate it from the data.
- We choose to predict the motion field using a Convolutional Neural Network (CNN) directly :
  - Difficulty : no direct supervision
  - requires the real motion field as target.

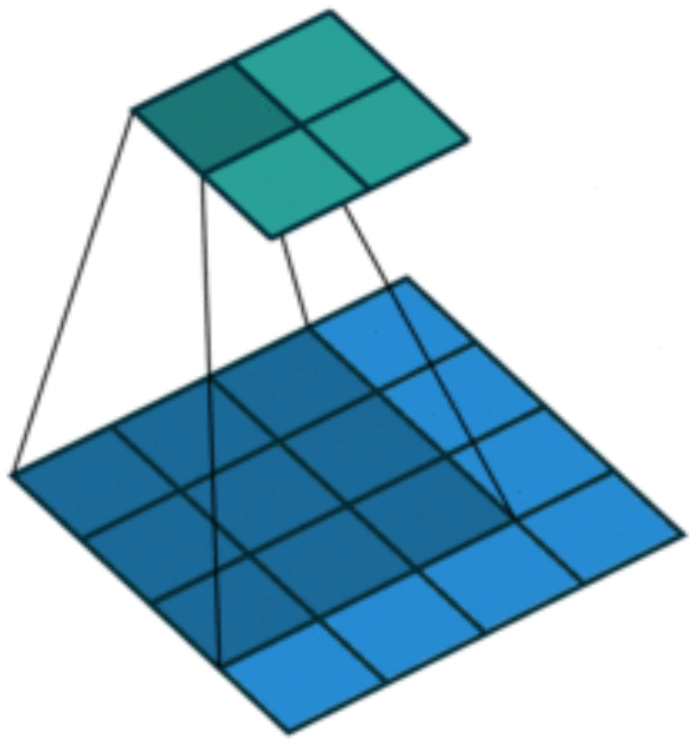
Convolutional Neural Network

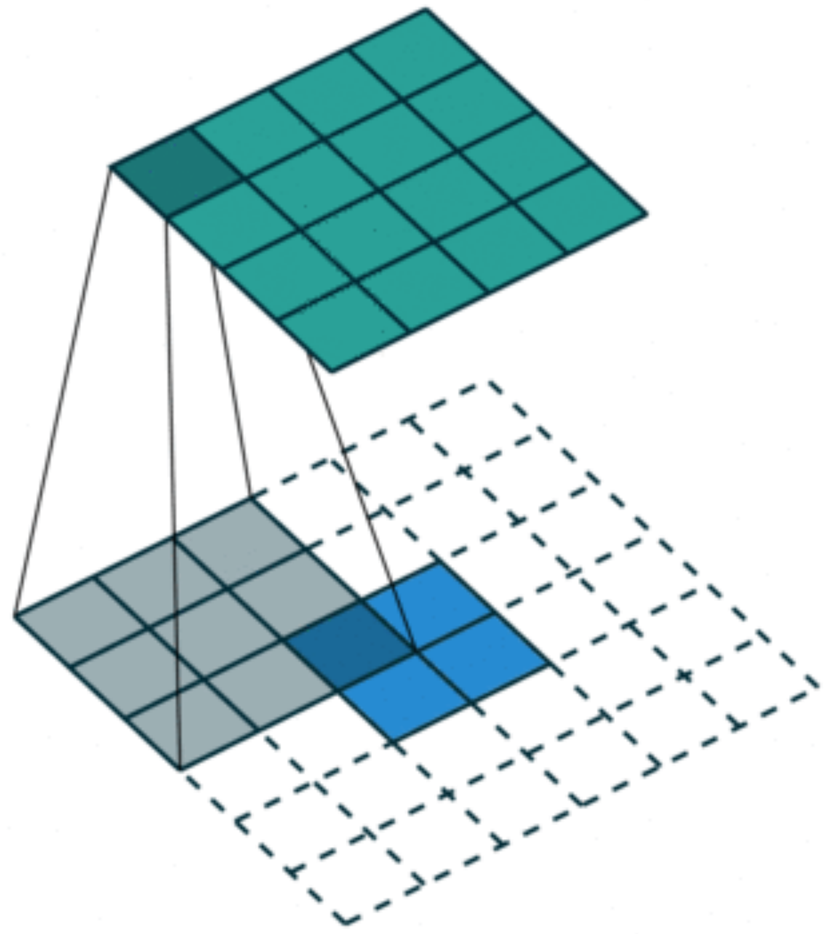


- **Classical CNN** : Take images as input, output a vector via dense layers.

- Useful for predicting a label, or a vectorial attribute.

Convolution - Deconvolution

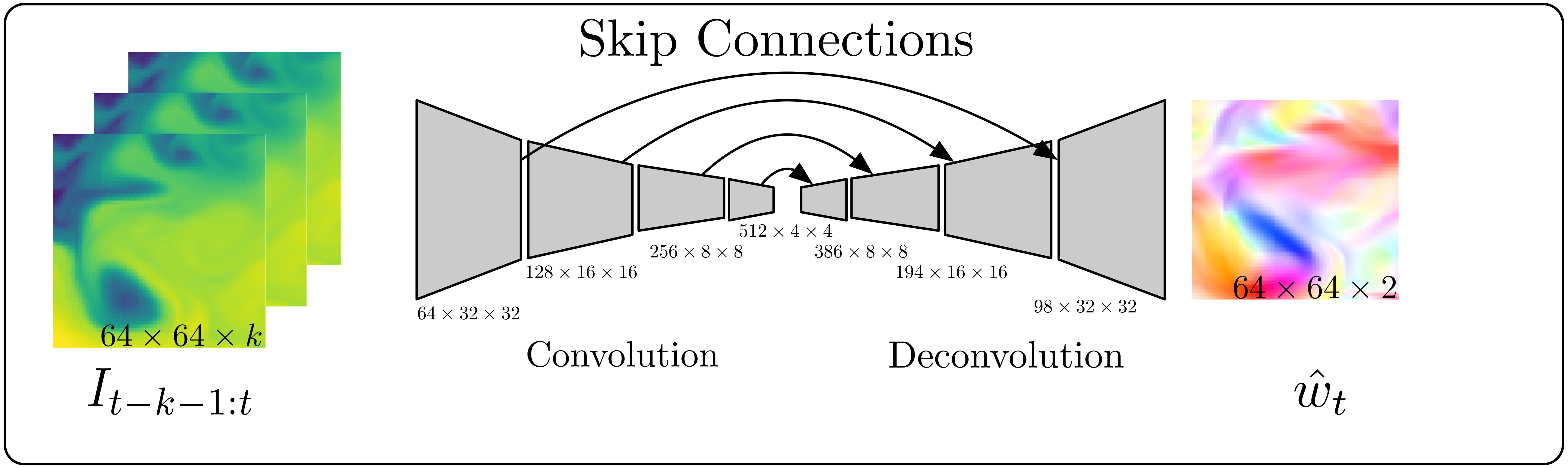




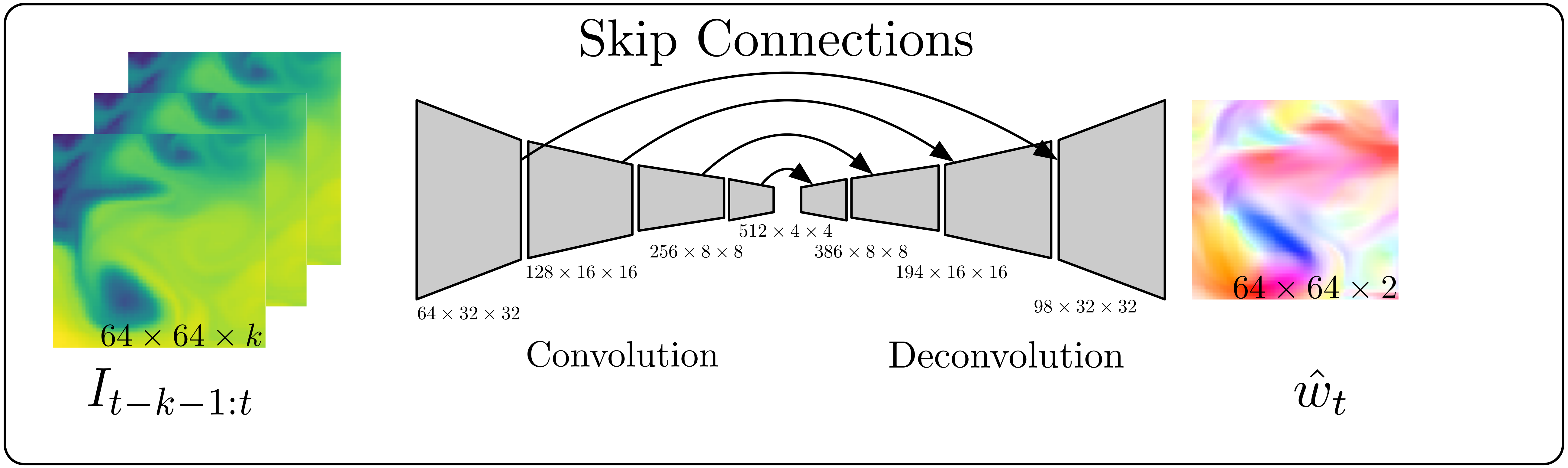
- **Convolution Deconvolutional NN** : Take images as input and output an images.

- Useful for forecasting, segmentation, etc…

Figure from V. Dumoulin

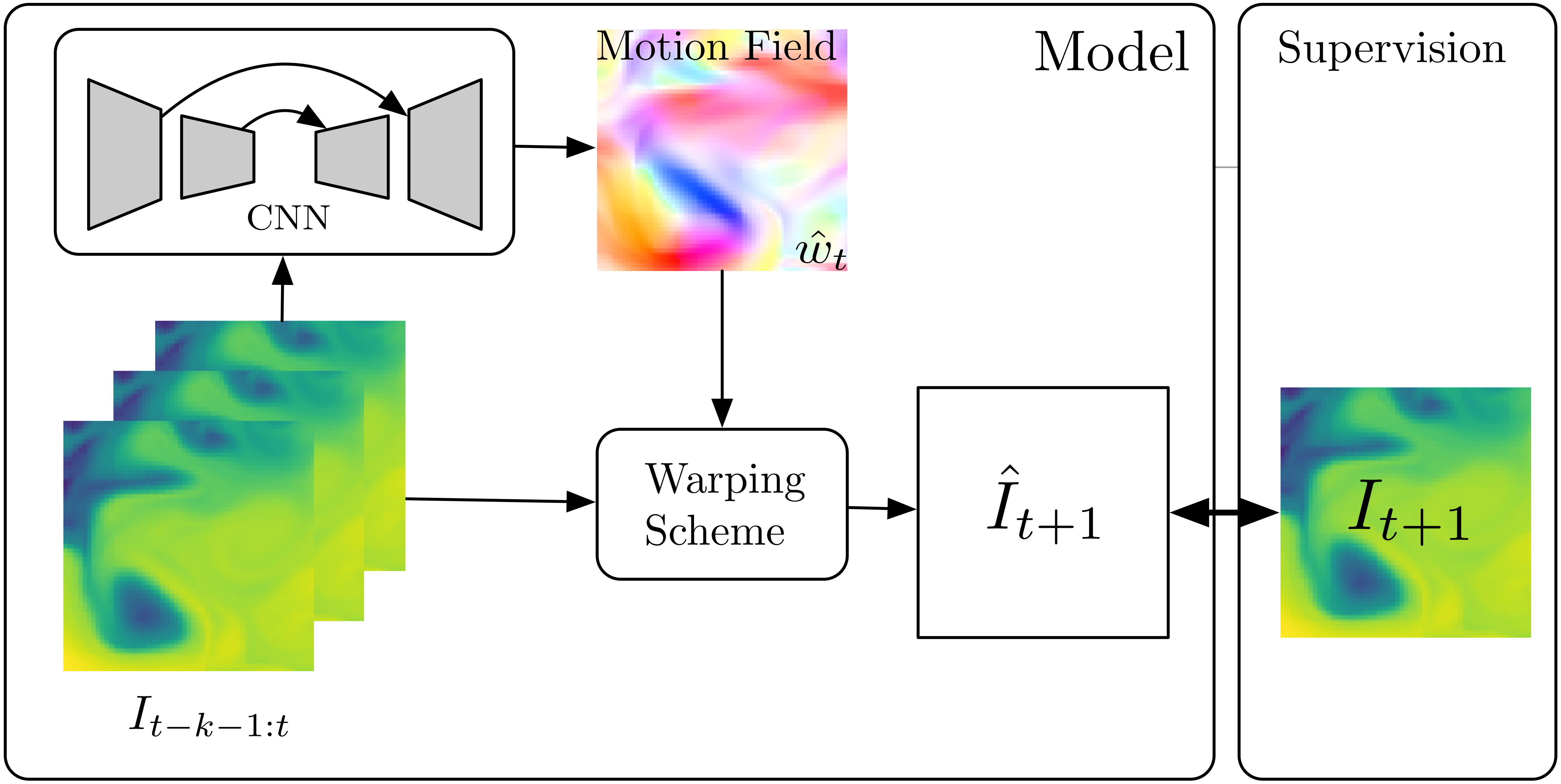


Our architecture



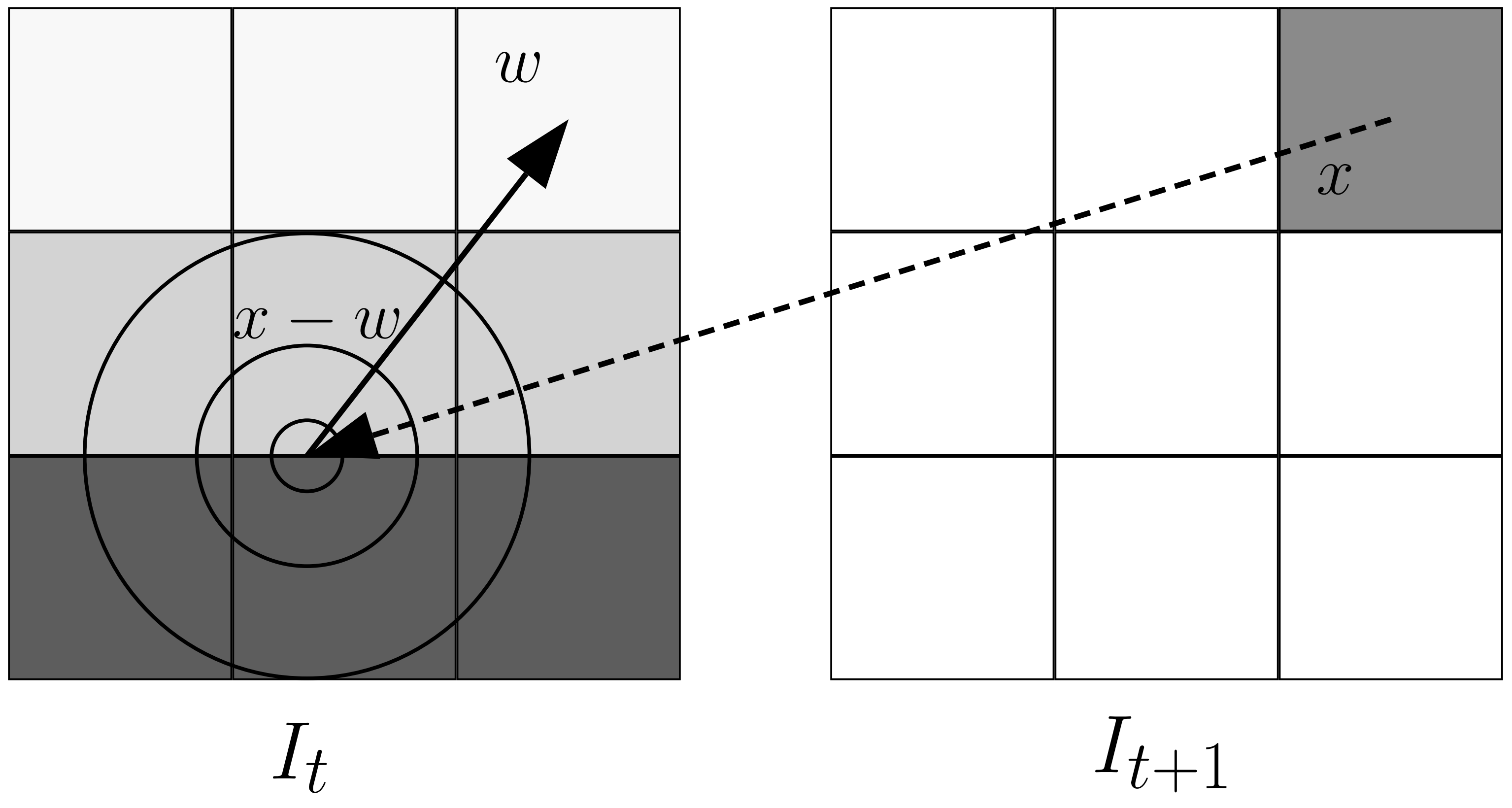
- We input past concatenated temperature images.
- The neural network estimates the displacement induced by the observation.

To train our NN motion estimator, we need the **target motion field,** which we usually don't have...



- We predict the next image with the motion field via a differentiable **warping** scheme
- The model is then **supervised** by the target image.

Warping

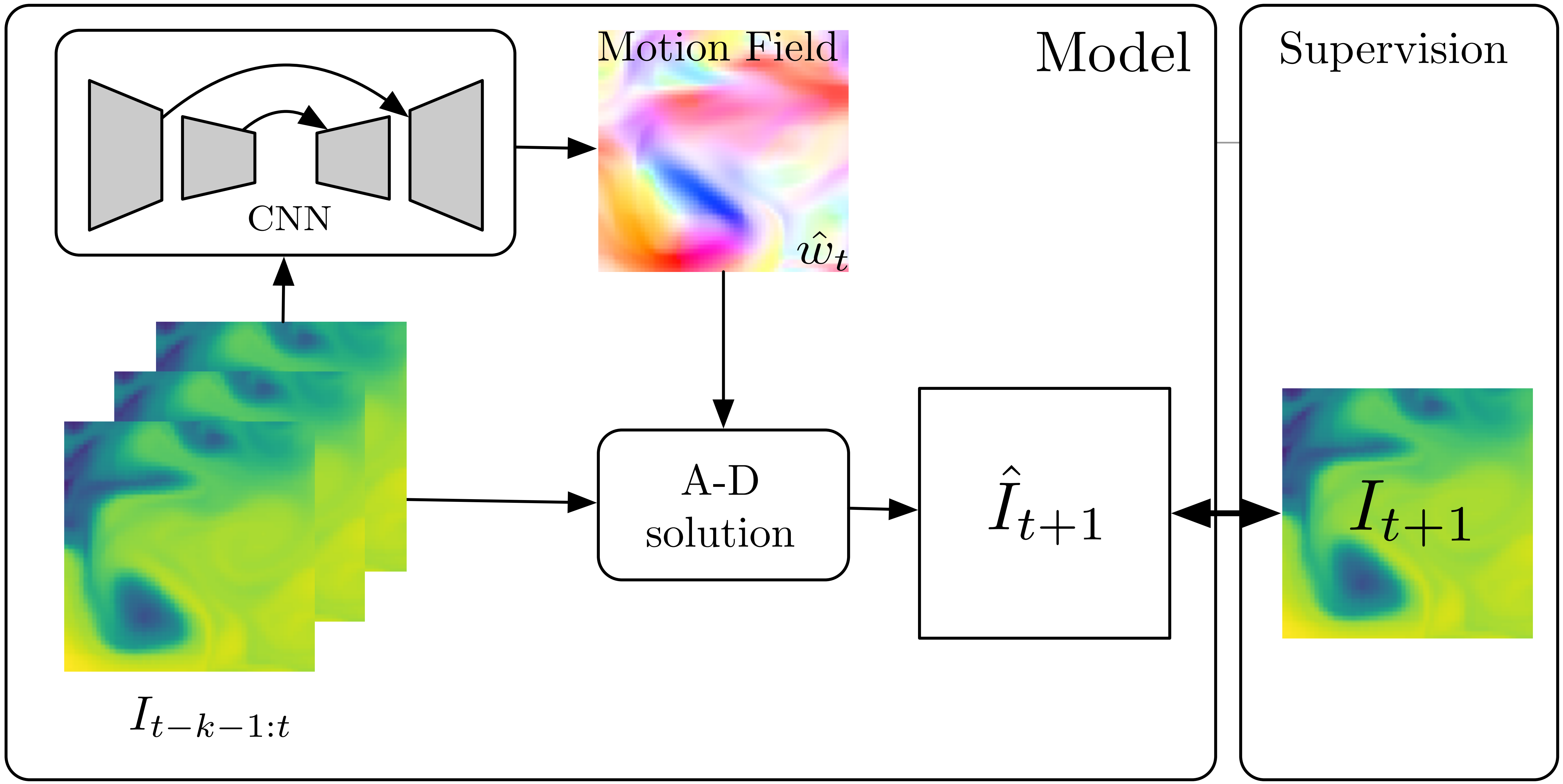


$$\widehat{I}_{t+1}(x) = \int_{\Omega} k(\, x - \hat{w}(x),\; y)\; I_t(y)\, dy$$

$$\approx \sum_{y \in \Omega \cap \mathbb{Z}^2} k(x - \hat{w}(x), y)\; I_t(y)$$

With $\quad k(x - w, y) = \mathcal{N}(y \,|\, x - w,\; 2Dt)$

This formulation is similar to the one of the Differentiable Image Sampling Scheme of the Spatial Transformer Network (Jadelbers, 2015)



Model
Supervision
Motion Field
CNN
$\hat{w}_t$
A-D solution
$\hat{I}_{t+1}$
I_{t+1}
$I_{t-k-1:t}$

Our model

Adding more Prior Knowledge

Cost Function for a single example :

$$L_t = \sum_{x \in \Omega} (\hat{I}_{t+1}(x) - I_{t+1}(x))^2$$

It is possible to add specific weighted penalties to the cost function representing our prior knowledge about *w* (divergence, smoothness, magnitude) :

$$L_t = L_t + \sum_{x \in \Omega} \lambda_{\text{div}}(\nabla .\, \hat{w}_t(x))^2 + \lambda_{\text{grad}} \left\| \nabla \hat{w}_t(x) \right\|^2 + ...$$

Recap

$$\frac{\partial w}{\partial t} = F(w)$$

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I$$

$$\nabla.w = 0$$

$$\ldots$$

Recap

$$\frac{\partial w}{\partial t} = F(w)$$

Directly integrated in the NN architecture.

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I$$

$$\nabla.w = 0$$

$$\dots$$

Recap

F is approximated with a CDNN

Directly integrated in the NN architecture.

$$\frac{\partial w}{\partial t} = F(w)$$

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I$$

$$\nabla.w = 0$$

$$\ldots$$

Recap

F is approximated with a CDNN

$$\frac{\partial w}{\partial t} = F(w)$$

Directly integrated in the NN architecture.

$$\frac{\partial I}{\partial t} + (w.\nabla)I = D\nabla^2 I$$

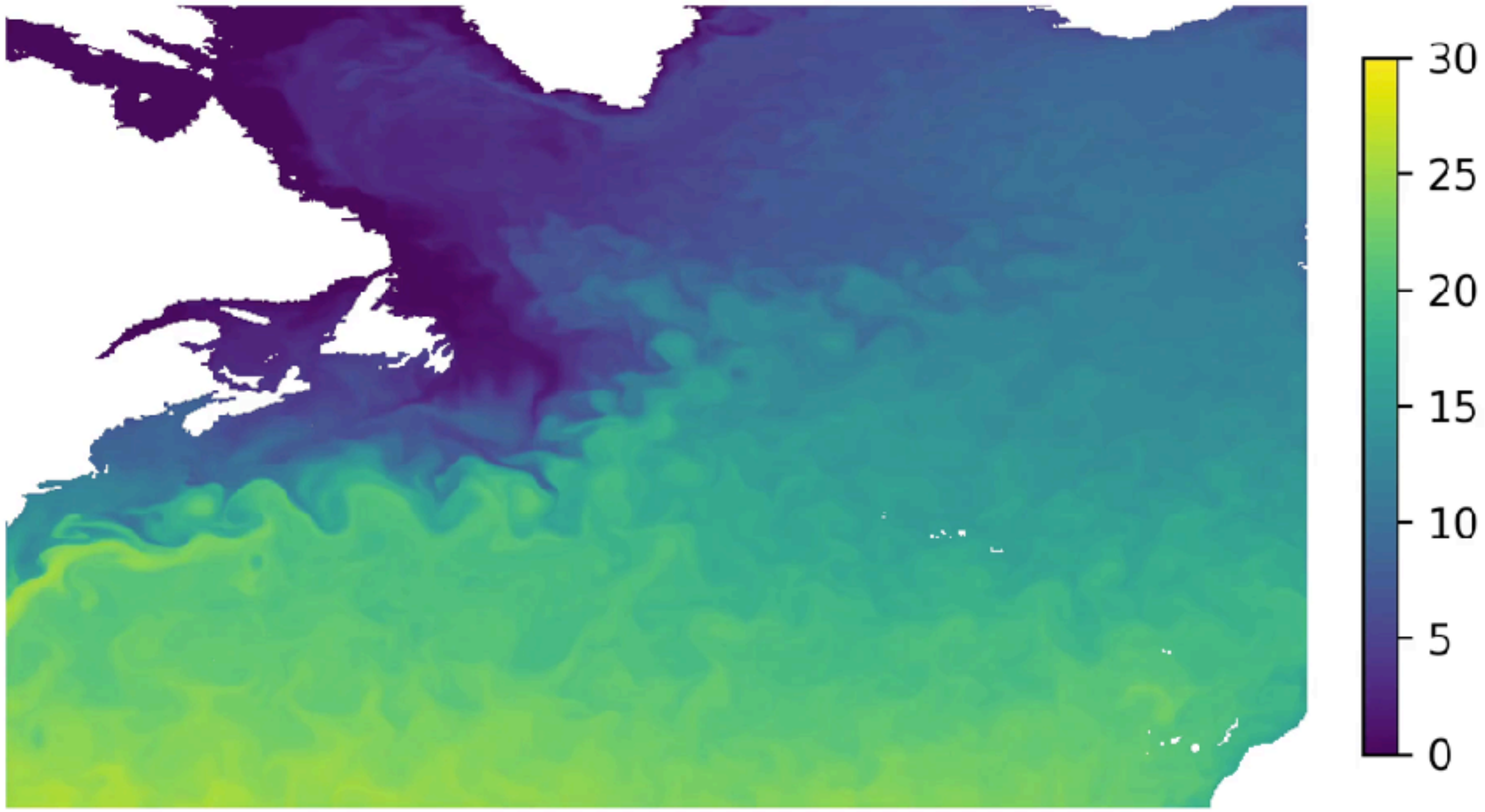
$$\nabla.w = 0$$

Integrated in the cost function with a penalty term.

$$\ldots$$

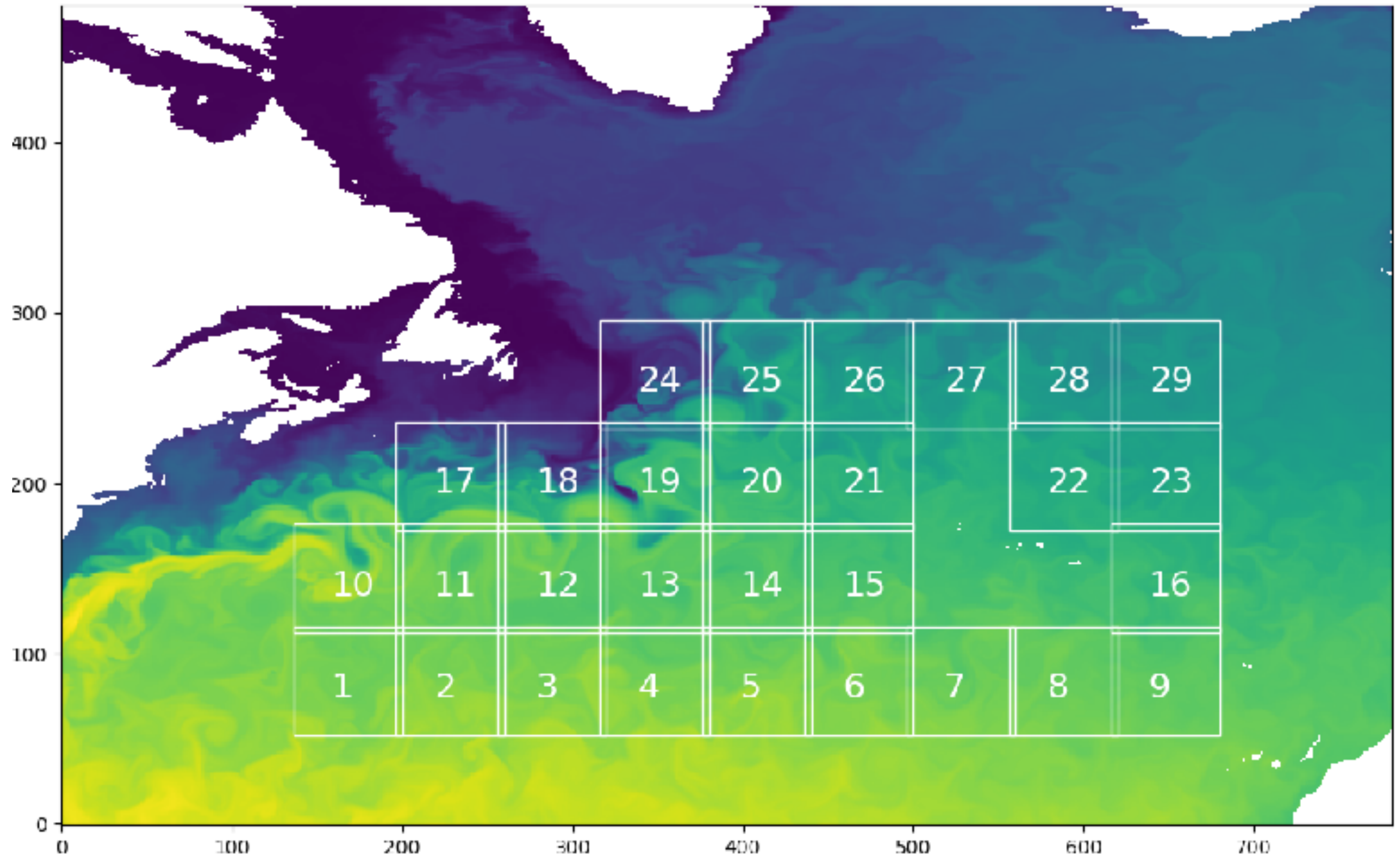
Experiments

Temperature: 2006-12-28



Data

- Synthetic (analyzed) data from the **NEMO** engine (Madec, 2008)
- We concentrate on 64 x 64 pixels subregions
- We use 2006-2015 for training and validation, and 2016 to 2017 for testing.



Baseline

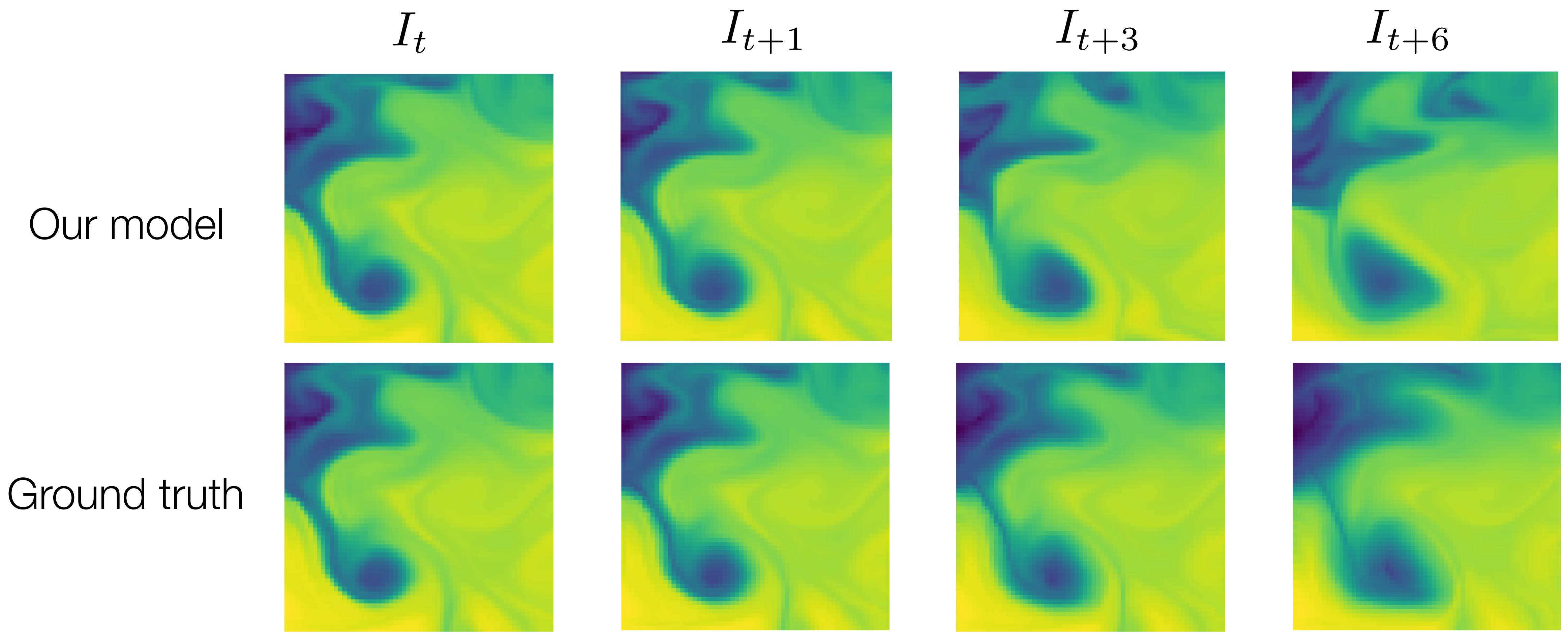
- Bereziat and Herlin (2015) is a numerical assimilation model which relies on data assimilation (shallow water equations). It is is a state of the art assimilation model for predicting ocean dynamics, here SST.

- Autoregressive CNN : an autoregressive convolutional-deconvolutional NN (ACNN), with an architecture similar to our Convolutionnal Deconvolutional module.

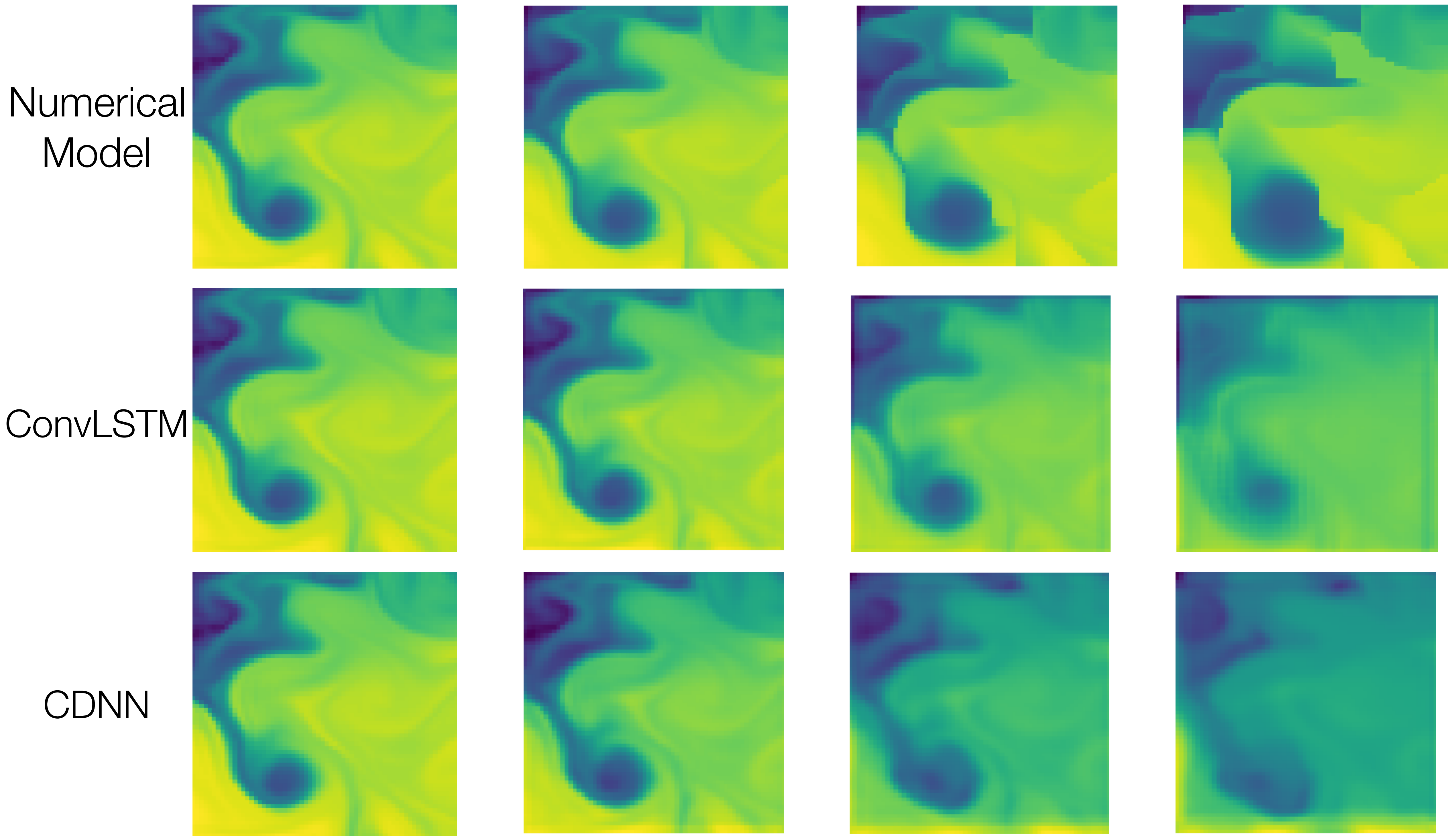
- ConvLSTM *(Shi, 2015)*, a Recurrent Neural Network which uses convolutional transitions in the inner LSTM module.

Results

Model	Average Score (MSE)	Average Time
Numerical model (Bereziat)	1.99	4.8 s
ConvLSTM	5.76	0.018 s
ACNN	7.84	0.05 s
Proposed model with regularization	**1.42**	0.040 s
Proposed model without regularization	2.01	0.040 s



Results

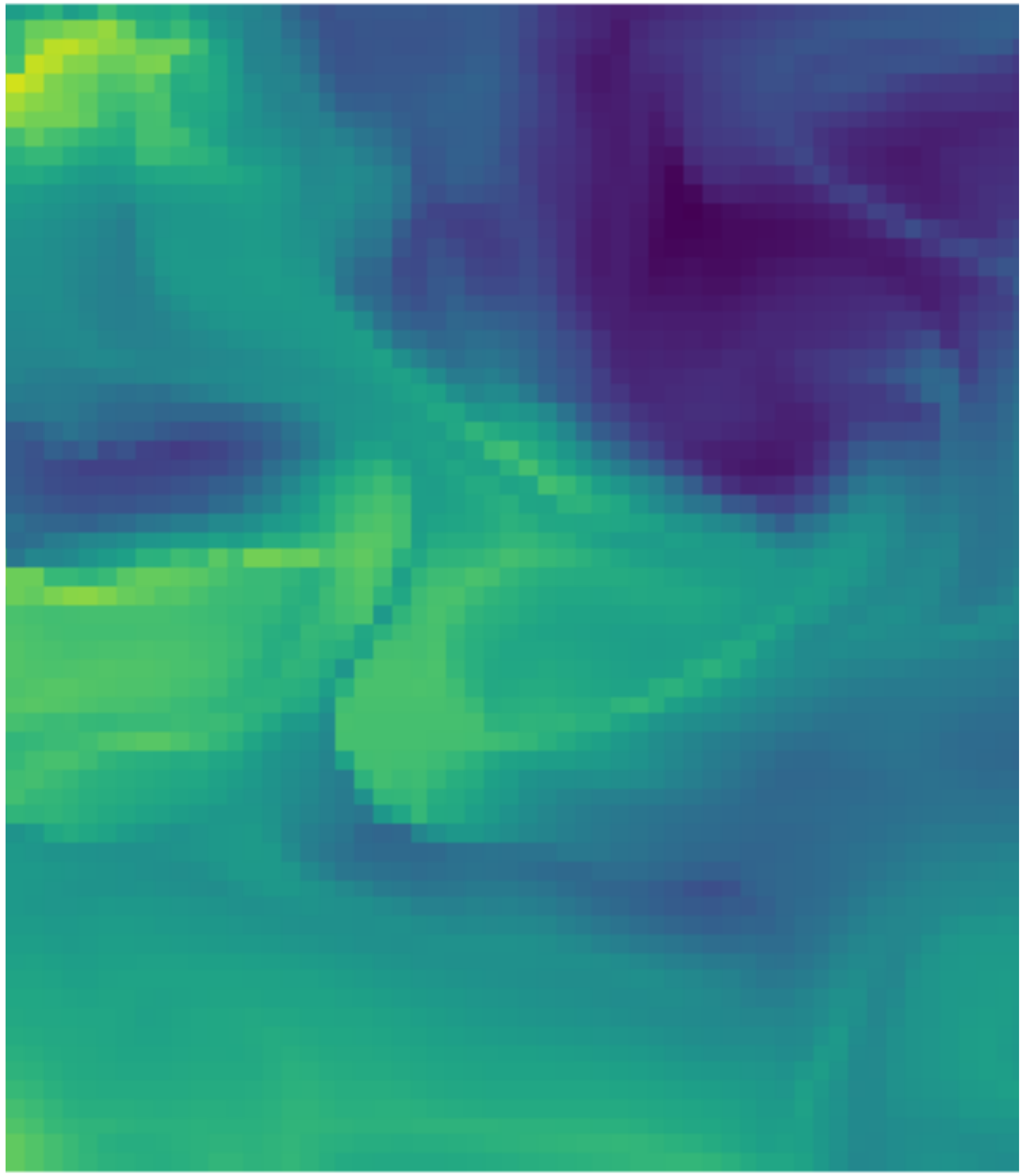


Conclusion

Two approaches

- **Physical Method :** can offer valuable intuitions for constructing Neural Network architectures.
- **Deep Learning ,** is an efficient data processing algorithm, with fast implementations, that allows us to develop and test new ideas quickly.

The interaction between the physical and the statistical paradigm for designing ***hybrid*** models is an interesting area of research to develop.



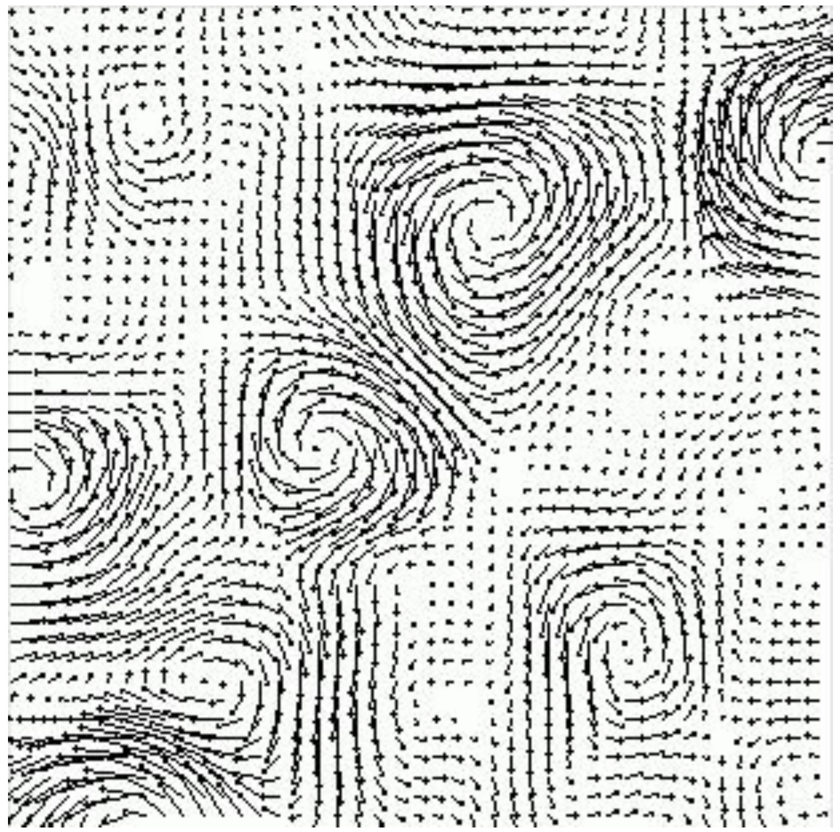
for more, see: arxiv.org/abs/1711.07970

But how can we estimate the motion field? w

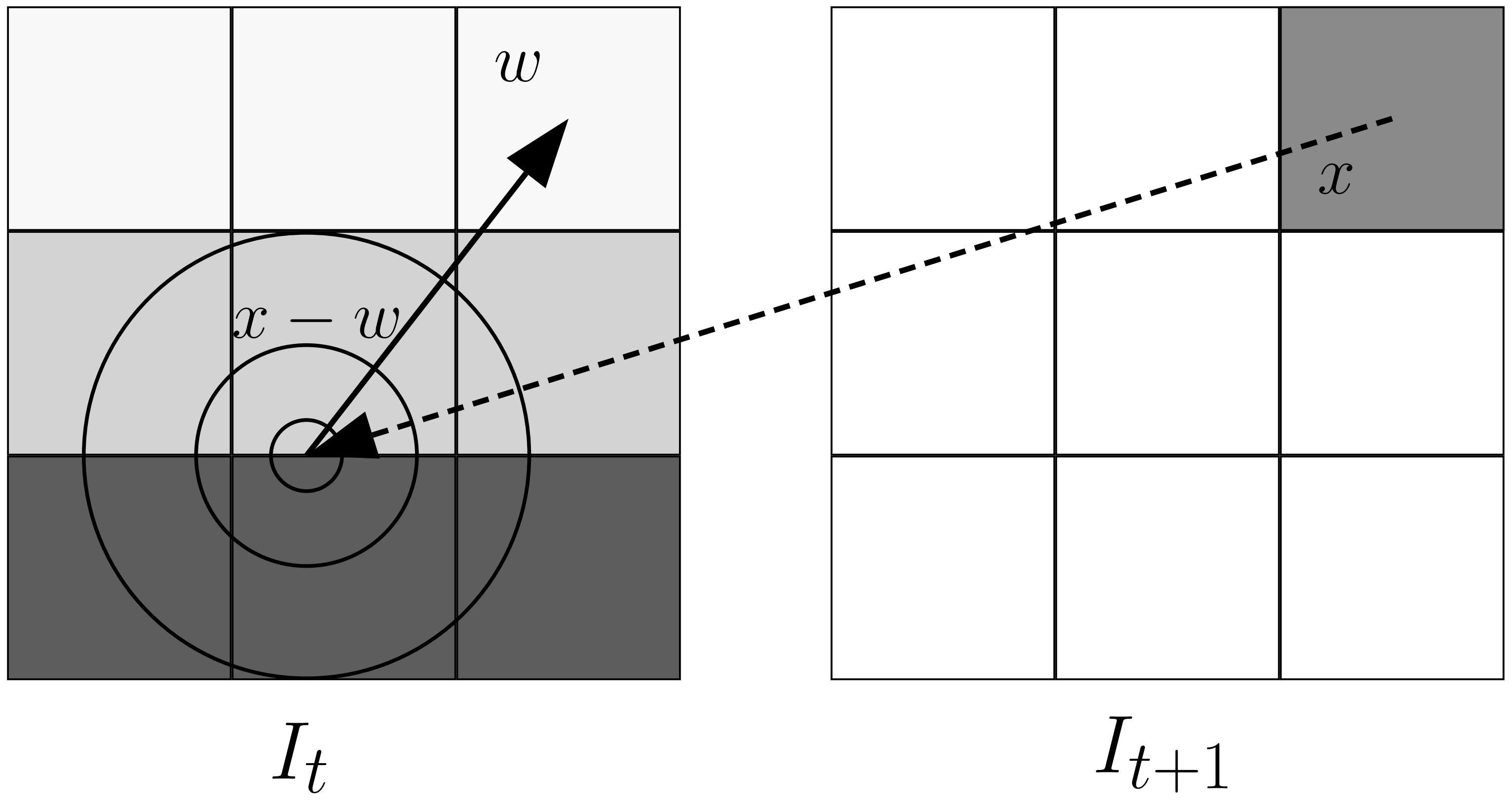
But how can we estimate the motion field? w

Our approach

- Similar to the physical approach
- We model the motion vector field explicitly.
- If we have an estimate of the motion vector, we can *displace* the temperatures along the motion estimate to forecast the temperatures.



Warping



The **Advection-Diffusion Equation** solution gives us a way to calculate the future images from past observation and the motion vector field :

$$\widehat{I}_{t+1}(x) = [f * I_t](x - \hat{w}(x))$$

Where f is a Gaussian pdf $f = \mathcal{N}(x \,|\, 0,\, 2D)$