

# Performing Particle Image Velocimetry using Artificial Neural Networks: a proof-of-concept

**Jean Rabault, Jostein Kolaas, and Atle Jensen**

Department of Mathematics, University of Oslo, N-0316 Oslo

Corr. auth.: jeanra@math.uio.no

**Abstract.** Traditional programs based on feature engineering are under performing on a steadily increasing number of tasks compared with Artificial Neural Networks (ANNs), in particular for image analysis. Image analysis is widely used in Fluid Mechanics when performing Particle Image Velocimetry (PIV) and Particle Tracking Velocimetry (PTV), and therefore it is natural to test the ability of ANNs to perform such tasks. We report for the first time the use of Convolutional Neural Networks (CNNs) and Fully Connected Neural Networks (FCNNs) for performing end-to-end PIV. Realistic synthetic images are used for training the networks and several synthetic test cases are used to assess the quality of each network predictions and compare them with state-of-the-art PIV softwares. In addition, we present tests on real-world data that prove that ANNs can be used not only with synthetic images but also with more noisy, imperfect images obtained in a real experimental setup. While the ANNs we present have slightly higher Root Mean Square (RMS) error than state-of-the-art cross-correlation methods, they perform better near edges and allow for higher spatial resolution than such methods. In addition, it is likely that one could with further work develop ANNs which perform better than the proof-of-concept we offer.

## 1. Introduction

Since the diffusion of ideas and methods related to Artificial Neural Networks (ANNs) into Fluid Mechanics is still limited at the time of writing this article [26], a brief general introduction about ANNs is included. More detailed introductions are available in articles and books [15, 41, 28, 9].

Neural networks are the attempt to reproduce in machines some of the features that are believed to be at the origin of the intelligent thinking of the brain [28]. The key idea consists in performing computations using a network of simple processing units, called neurons. The output value of each neuron is obtained by applying a transfer function on the weighted sum of its inputs [9]. When performing supervised learning an algorithm, such as stochastic gradient descent, is then used for tuning the neurons weights so as to minimize a cost function on a training set [9].

Attempts to develop ANNs appeared with the first computers [37], but with limited success until recently. Looking back at the history of neural networks development, it appears that many of the key ideas had been present in a long time, but that both computational power and semi-empirical best practice rules were lacking until recently. For example the

importance of moving to Rectified Linear Units (ReLU) to avoid artificial neurons saturation, rather than using sigmoidal neurons that are better approximation of biological neurons, was understood only recently compared with the age of the field [8]. Similarly the use of backward propagation of the error gradient together with Convolutional layers was proposed only in the 1990s [29] and was widely adopted even more recently [25, 28]. In the same vein, a simple systematic method for selecting reasonable variance for the initialization of neural network parameters taking into account the size of both the upstream and downstream layers was presented as late as year 2010 [7], even though a simpler method based on the same idea was already presented in 1998 [30]. In addition the computational power available limited until recently large scale use of neural networks to relatively simple tasks, such as digits recognition [29]. While it is well known that a large enough feed-forward neural network can fit arbitrarily well any function and that feed-forward neural networks are therefore universal approximators [17], or that the recurrent neural network paradigm is Turing complete [39], the proofs of those results do not predict anything about the size, architecture or training method that should be used to build and optimize those networks. Therefore, designing neural networks is largely an experimental science: one starts with a simple network (which architecture is critical for performance, and is often chosen following currently available best practice semi-empirical rules), and increases its complexity until over-fitting occurs or limits in the computing power available make it impossible to further increase the model size.

Despite those difficulties many breakthroughs have been achieved in the recent years. The first step in the renewed interest for neural networks came in 2012 when it was shown that using Convolutional Neural Networks (CNNs) could reduce the error rate in an image classification task by a factor of two compared with the best feature engineering methods available [25]. Following this milestone, it has become clear that feature engineering underperforms compared with neural networks in a variety of tasks including image classification, speech and hand writing recognition, text analysis, and control of autonomous cars among others.

Experimental Fluid Mechanics relies on using image processing for measuring flow velocities. Particle Image Velocimetry (PIV) and Particle Tracking Velocimetry (PTV) are two popular measurement techniques that rely on comparing images of a flow seeded with tracer particles and separated by a short time interval, in order to gain information about the flow motion [36]. This makes it possible to reconstruct a velocity field (PIV, [1, 42]), or to track the motion of individual particle images between pictures (PTV, [3]), or a combination of both [20]. A variety of techniques can be used to implement each of those methods. PIV processing usually relies on computing the cross-correlation of a spatial window between two frames for finding a correlation peak, which indicates the most probable displacement of the flow in the corresponding window [42, 13], but several other methods and algorithms were presented in the literature [38, 35, 5, 12]. PIV algorithms have been refined and complexified with time so that several correlation techniques can be used, subpixel accuracy can be achieved, and outliers can be automatically detected and interpolated [42, 13, 33]. PTV processing on the other hand relies on tracking individual particle images. The particle image centres are usually identified using a blob algorithm [3], and a pairing of the particle image centres identified in

the different frames is then performed by minimizing a cost function [3].

On many aspects, the methods currently used for performing PIV and PTV rely on complex feature engineered algorithms. Based on the trends observed in the other branches of image processing, one can therefore expect that well designed ANNs should become better at performing those tasks than the algorithms used today. In addition, ANNs could be a solution to some of the limitations of current PIV algorithms. In particular, ANNs can be trained to evaluate velocity gradients from a simple subwindow, which is not easy with area based matching algorithms [43] including current PIV algorithms, though some attempts have been made [6]. ANNs have been suggested in the past for tackling PIV and PTV related problems, but to the authors knowledge all the corresponding articles date back to before the recent improvements in the understanding of ANNs and therefore the use of ANNs for performing PIV should be investigated again, following the technical improvements that have emerged recently. In addition, ANNs were mostly investigated for performing only parts of the PIV or PTV processing in complement of traditional methods, rather than as a standalone method. Such uses include detection of spurious modes in conventional PIV algorithms output [31], image denoising [11], identification of particle image centres [11], or pairing of particle image centres for PTV tracking [10, 14, 27].

In the present article, we investigate how ANNs built following some of the recent best practice design rules perform at extracting flow velocity from a pair of PIV images. Both a Convolutional Neural Network (CNN) and a Fully Connected Neural Network (FCNN) are evaluated. Synthetic images representative of real PIV data are used for both training the networks and benchmarking against state-of-the-art PIV codes. In addition, we perform a benchmarking on some real-world data, which proves that ANNs can be used not only with synthetic images but also with more noisy, imperfect images obtained in a real experimental setup. In section II we describe the architecture of the ANNs implemented and the method used for generating the synthetic PIV pictures. In section III we describe the conventional PIV technique used as a reference for comparison with ANNs. In section IV we analyze the results obtained and compare them with state-of-the-art PIV codes. Finally we discuss our results and further work.

## 2. Training set and Neural Networks used

The synthetic data used for training and evaluation are generated in Matlab. Tensorflow is used for performing training and evaluation of the models. GTX970 and GTX980 TI GPUs performing single precision computations are used in all the following. For both networks, images are fed to the network by batches of 128 pairs. A pair of images is considered as a two channel input, so that the input dimension corresponding to one batch is  $128 \times 32 \times 32 \times 2$  pixels. Both neural networks are trained to predict the linear displacement along the X and Y axis and the Jacobian deformation matrix, i.e. 6 quantities in total.

In this section, we first describe the procedure applied for generating synthetic images before presenting the architecture of each ANN used.

### 2.1. Synthetic training set

Synthetic data [21] are used for training and evaluation of the networks. This allows to create arbitrarily big labeled training dataset and therefore issues common with ANN training, such as over-fitting, are avoided. Each image pair is generated independently from the rest of the dataset. Each time a new image pair is generated, second order polynomials are created for the  $u$  and  $v$  components of the velocity as:

$$\begin{cases} u(x,y) = U_0 + \bar{J}_u \cdot \bar{r} + \bar{r}^T \cdot \bar{H}_u \cdot \bar{r} \\ v(x,y) = V_0 + \bar{J}_v \cdot \bar{r} + \bar{r}^T \cdot \bar{H}_v \cdot \bar{r}, \end{cases} \quad (1)$$

where the position of the point considered is  $\bar{r} = (x, y)$ ,  $(U_0, V_0)$  is the velocity at the centre of the image, and the Jacobian and Hessian tensors of the velocity component  $i$  at the centre of the image are  $\bar{J}_i$  and  $\bar{H}_i$ , respectively.  $U_0, V_0, \bar{J}_i$  and  $\bar{H}_i$  are randomly generated from the uniform distribution.  $U_0$  and  $V_0$  are in the range  $\pm 4 \text{ pixels/frame}$ ,  $J_i$  in the range  $\pm 0.05/\text{frame}$ , and  $H_i$  in the range  $\pm 0.001/\text{pixels frame}$ . Those values are typically representative of real-world applications, even though training on a wider pixel displacement range would be necessary to go beyond a proof-of-concept. The maximum pixel displacement value was chosen so that reasonably fast training time could be achieved, when we were still exploring ANN designs but did not know if end-to-end PIV could even be performed. However extending to wider displacements presents no theoretical difficulty, as it is simply a matter of generating a larger training dataset, featuring also images with more important pixel displacements. For each image pair  $U_0, V_0, \bar{J}_u$  and  $\bar{J}_v$  are concatenated into a label vector. Therefore the neural networks get trained to predict not only the translation velocity at the centre of the images, but also the velocity gradients of the image.

Once the random velocity field has been generated, a set of initial positions for the tracer particles is drawn from a random uniform distribution. The particles are assumed to follow perfectly the flow so that the equation describing their advection by the velocity field is:

$$\frac{d\bar{x}_p}{dt} = \bar{u}(\bar{x}_p(\bar{x}_0, t), t), \quad (2)$$

with  $\bar{x}_p(\bar{x}_0, t)$  the position of the particle  $p$  initially at the position  $\bar{x}_0$  after a time  $t$ . Equation (2) is then integrated half a timestep backward and forward in time using the Runge-Kutta 4 method for generating the particles positions in both frames of the image pair. The velocity field and particle distribution are extended to an area slightly larger than the size of the image, allowing the particles to leave and enter the field of view.

Sufficiently small particles imaged by a camera form circular patterns known as Airy disks. The central lobe of the Airy disks are normally well approximated by Gaussian bell curves. Gaussian distributions are therefore used to generate the synthetic particle images [36]:

$$I(x,y) = I_0 \exp\left(\frac{-(x-x_0)^2 - (y-y_0)^2}{(1/8)d_\tau^2}\right), \quad (3)$$

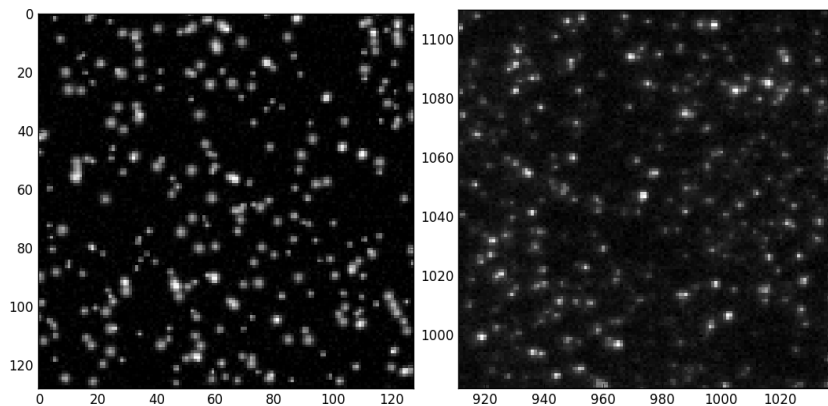


Figure 1: Comparison between a  $128 \times 128$  pixels artificial image (left) and a real image sample of similar size (right).

where  $I_0$  is the particle image luminosity,  $d_\tau$  the effective particle image diameter, and  $(x_0, y_0)$  the position of the particle image centre.  $I_0$  and  $d_\tau$  are random and drawn from a uniform distribution independently for each particle image, resulting in typical particle images size of 2.5 pixels (varying between 1.5 and 3.5). The particle image intensity is not integrated over the pixel area in order to save computation time, which may introduce a small source of noise that is probably negligible owing to the size of the particles.

As a final step a Gaussian white noise of variance 1% of the maximum image intensity is added to the images. The aim of this Gaussian noise is to train the network on non-perfect data, in order to make the training set closer to reality and the training more robust. 1% is taken as a proxy value, which is deemed reasonable in the case of an experiment. A comparison between a  $128 \times 128$  pixels artificial image and a real image sample of similar size is presented in Fig. 1.

## 2.2. Convolutional Neural Network

Several variants of CNNs were tried before fine training the best prototype. The best prototype convolutional network finally used is composed of a single convolution layer featuring 512 kernels of size  $16 \times 16$  pixels and depth 2 applied with a stride of 8 pixels (as a consequence, no zero padding is needed), so that the size out of the convolution layer is 8192, followed by fully connected layers. The two images being fed in the network are considered as two channels. Four fully connected layers are used to process the data generated by the convolutions. The sizes of the fully connected layers are 8192, 4096, 2048 and 6 neurons going downwards in the network. The first three layers use leaky Rectified Linear Units (leaky ReLUs) of slope 0.1 for negative  $x$  values. The last layer uses linear activation function for producing the output prediction of the network. The whole network is trained using the Adam optimizer. Gradient Descent, Adadelta and Adagrad optimizers were tested on an early version of the network but found to perform less well. Dropout layers were used in the early versions of the network but removed later on since arbitrarily much data can be generated to

train the network, eliminating the over-fitting issue and the need for regularization. Similarly, while  $L^1$  and  $L^2$  regularization were tested on early versions of the network, they were not included in its final version. The cost function to be minimized is the absolute norm of the prediction error:

$$Cost = |label - prediction|. \quad (4)$$

$L^2$  norm was tested on early versions of the model but found less performant. The Xavier initialization relying on a Normal distribution [7] is used, except for the convolutional layers where a reduction factor is further applied to the Xavier standard deviation value. The learning rate is progressively decreased following an exponential decay rate scheme, and the absence of over-fitting is checked by comparing the prediction error on the training set with the one on a separate test set. Both the training set and the separate test set are generated from the exact same random images generation algorithm previously presented, so that this is formally equivalent to using a validation data split.

There are several reasons why CNNs are appealing for performing PIV. The use of convolution kernels, that are applied in the same way on several parts of the image pair, makes sure that a similar processing is applied on all parts of the image. It is easy to understand why convolution kernels can perform tasks that are relevant to PIV. For example, the following  $3 \times 3 \times 2$  kernel:

$$K_{:, :, 0} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, K_{:, :, 1} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \quad (5)$$

where  $K_{:, :, 0}$  is the kernel slice applied on the first image (first channel) and  $K_{:, :, 1}$  the kernel slice applied on the second image (second channel), computes the local difference between the image channel 1, and the channel 2 translated by one pixel along x direction. This difference can then be interpreted by the fully connected layers of the network. While it is not possible to know which processing strategy the fully connected layers of the network choose, it would be possible for them to compute for example a  $L^1$  or  $L^2$  norm of the difference. Methods based on the  $L^1$  or  $L^2$  norm of the difference between translated windows, while less common than the cross-correlation approach, can be used to perform PIV [32] and are available in for example Digiflow [2].

We can further investigate the strategy followed by the convolution layer by studying in details the convolution kernels obtained. In many ANN applications, including the case of object recognition, it is well established that convolution kernels ideally should evolve into feature extractors as the network properly converges, leading to well defined shapes to be visible in the kernels [25]. However, in the present case, the data to analyse has much less structure than what is present in a typical object recognition or image analysis task: the most apparent property of a single image, which is the particle images distribution, is random and the information to be extracted is contained in the time evolution of this random pattern.

Convolution kernels are shown in the two first lines of Fig. 2. Very little structure is visible. While this could be the sign of poor network convergence, we carefully checked

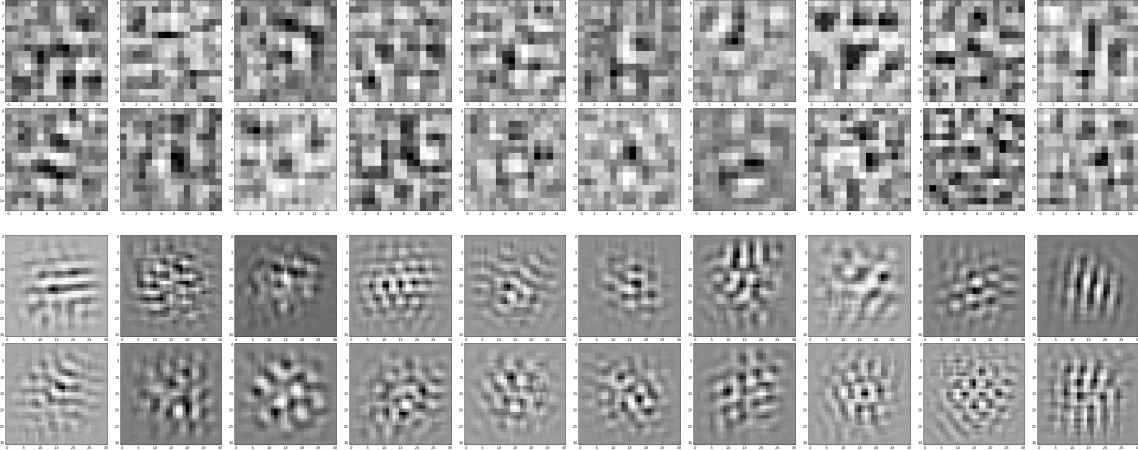


Figure 2: Top two lines: first channel of the convolution kernels 0 to 20. The kernels seem messy and no clear global structure is visible. This sample is representative of what can be observed for all 512 kernels. Bottom two lines: cross-correlation maps computed between the first and second channel of each convolution kernel (we reproduce here only the first 20 such correlation maps). The cross-correlation maps display some clearly visible patterns. This sample is representative of what can be observed on the cross-correlation maps of all 512 kernels.

that the network convergence is, if not finished, at least extremely slow by training the networks for times several times longer than what is required to reach a plateau in prediction performance, and also verifying that diminishing or increasing the learning rate does not yield better training. The absence of structure in the individual channels of each kernel may not be surprising even for an otherwise well-converged network, as the pattern present in each individual image is completely random. As a consequence, the shape of one kernel channel may not be important by itself, but in relation to the shape of the other channel of the same kernel. Therefore, we compute the cross-correlation between the two channels of each kernel, defined in our case as:

$$C_i(k,l) = \sum_{m=0..W-1} \sum_{n=0..W-1} K[m,n,0,i]K[m-k,n-l,1,i], \quad (6)$$

where  $W = 16$  is the kernel width,  $i$  is the number of the two-channels kernel considered, 0 or 1 the channel considered,  $-(W-1) < k, l < (W-1)$ , and the value of a kernel channel taken outside of its domain of definition is set to zero.

Results are presented in the two lowest lines of Fig. 2. Structures clearly appear in the cross correlation between the two channels of each kernel. The patterns are concentrated close to the center of the correlation maps, which is expected as the zero padding introduced in the computation of the cross-correlation maps smears out results obtained for large negative and positive  $k$  and  $l$  indexes.

It is worth mentioning that, since the convolution operation is linear, the output of a convolution kernel over the whole picture can be recovered from the sum of the output of the

same convolution kernel on several parts of the pictures pair. Therefore the use of kernels smaller than the size of the images does not restricts the ability of the network to compute a pattern difference over the whole picture, since this information can be obtained from summing the different outputs of one kernel over the pictures.

A limitation with convolution kernels is that their shape cannot get adapted to the flow gradients, since their shape is fixed a priori when performing a computation. By contrast, state-of-the-art PIV codes use window deformation to take into account the fact that an initially square flow area gets distorted by velocity gradients into a more complex shape as time increases. This effect is then taken into account by performing multi-pass processing, where the shape of the second window is computed based on the velocity and velocity gradients obtained from the previous flow field estimates [18, 19]. However, our CNN compares well in accuracy with such state-of-the-art method as shown in section 4.

### 2.3. Fully Connected Neural Network

Several configurations of FCNNs consisting of fully connected leaky ReLUs were evaluated. The data corresponding to the two pictures are directly injected into the first fully connected layer. The best prototype network has a total of 6 layers. The first five layers feature 4096 leaky ReLUs, and the last layer has 6 linear units that are used to produce the network prediction. The leaky ReLUs have a slope 0.1 for negative  $x$  values, the Adam optimizer is used to minimize the absolute norm of the prediction error and no regularization is imposed on the network. The Xavier initialization relying on a Normal distribution [7] is used on all layers. The learning strategy is similar to what was presented for the CNN.

## 3. Particle image velocimetry setup

In all the following, comparisons were performed against LaVision DaVis v8.1.1, a market leading commercial code, and HydrolabPIV, an in-house PIV code developed at the University of Oslo [22].

State-of-the-art PIV algorithms are based on multipass methods and therefore designed to work on full size images, which makes a direct comparison of subwindow performance with ANNs irrelevant. To overcome this problem, traditional PIV codes were used on extended images of size 128x128 pixels so that multipass can be applied, and the velocity from the center 32x32 subwindow was extracted for generating the predictions of the ANNs. Since the predictions of the ANNs are performed based on the single 32x32 center subwindow, this gives an advantage to the classical PIV codes as they are given sixteen times more pixels. Of course, one cannot assert that the traditional PIV method uses sixteen times more information than the ANNs, since information far from the center of the image is less predictive of the velocity at this position. Based on the maximum subwindow size used, the overlap value selected, the extent of B-spline basis functions used to approximate the velocity field for window distortion and to some degree the width of the Lanczos kernel, we estimate that the HydrolabPIV code uses in practice slightly over four times more information than the ANNs



for predicting the velocity at the center of the image. This is likely similar for LaVision, where few details of the computational strategies are available.

### 3.1. LaVision

The PIV processing performed in LaVision was done using multpass with decreasing interrogation windows size and pixelwise windows deformation. A total of four passes were used, which consisted of two passes with 64x64 pixels subwindows, followed by two passes with 32x32 pixels subwindows. A 75% overlap was used for all passes. To achieve the best possible quality, normalized cross-correlation was used in all passes and the LaVision high accuracy mode, which relies on Lanczos reconstruction for image interpolation, was enabled for the final pass.

### 3.2. HydrolabPIV

The PIV processing performed in HydrolabPIV was done using an initial pass with 48x48 pixels subwindows followed by a shifted pass with 32x32 pixels subwindows and finally four 32x32 pixels subwindow passes. All passes used masked normalized cross-correlations [34] and a 75% overlap. For the last four passes, windows distortion using Lanczos reconstruction were used.

## 4. Results

The quality of the predictions computed by the ANNs is evaluated using a set of test cases generated in a similar way as the training sets. As previously underlined the size of the test cases generated is 128x128 pixels so that conventional PIV methods can be used at the best of their performance for comparison, while reduced size 32x32 subwindows are given to the ANNs.

When the error should be evaluated on both components of the velocity prediction at the same time, the mean value of the Frobenius norm of the error is used:

$$\|u - \hat{u}\|_F = \sqrt{\frac{1}{N} \sum_{i=1}^N [(U_i - \hat{U}_i)^2 + (V_i - \hat{V}_i)^2]}, \quad (7)$$

where  $(U_i, V_i)$  is the  $i$ -th velocity prediction, and  $(\hat{U}_i, \hat{V}_i)$  is the true value used for generating the image. This error can be splitted in a variance error and a bias error, where the bias error is the mean value of the prediction error.

In the boundary performance tests the RMS error on the X component:

$$rmse(U) = \sqrt{\frac{1}{N} \sum_{i=1}^N (U_i - \hat{U}_i)^2}, \quad (8)$$

and the bias error on the X component:

$$\text{bias}(U) = \frac{1}{N} \sum_{i=1}^N (U_i - \hat{U}_i), \quad (9)$$

will be used instead.

#### 4.1. Simplified velocity fields

Several simplified velocity fields were used for comparing the quality of the results obtained from traditional PIV software with ANNs. Results are presented in Fig. 3.

The first test (Fig. 3 a), which is also the most common in the literature, is a pure translation test for which the velocity is constant within the image. While it is an easy test to implement, it is likely to underestimate the error in real world cases when gradients and curvatures are present. Both ANNs have a reasonable accuracy within the training range ( $\pm 4 \text{ pixels/frame}$ ). As expected the performance outside of the training range, indicated by the dashed line, is very poor. This could be easily improved by training the networks with an image set including greater maximum particles displacement. Both traditional PIV softwares are expected to do well for this test since their algorithms first assume that the velocity field within a subwindow is approximately constant, before refining the predictions through image deformation during the iterative multipass steps. We note that while HydrolabPIV has the lowest error, it also has some peak-locking effect, which is a bias error caused by subpixel interpolation.

In the second simplified velocity field test (Fig. 3 b) the velocity field is set to a constant gradient in the y direction, with a zero velocity at the center of the image. The overall RMS error value is similar to the pure translation case. However, rather than an abrupt change of the error when going outside of the training domain, the error obtained from the ANNs increases slowly as gradients become greater than the maximum value used during training. This means that the prediction of the velocity value is robust against local shear of the images.

In the last simplified velocity field test (Fig. 3 c) the velocity field is chosen to have a constant curvature rate in the y direction, while keeping the translation and local gradient values to zero at the center of the images. We note that the error is not constant within the training range, indicating a bias error which increases with the curvature. While the training set does have curvature, the ANNs are not trained to estimate the Hessian used for generating the images and therefore are not trained to recognize the effect of curvature in the images. Therefore we expect that the bias error due to the curvature would be likely to be reduced if the ANNs were trained predicting also the Hessian matrix, at the possible cost of a slightly increased RMS error. One can note that both traditional PIV codes also have a similar tendency towards bias error, though to a lesser extent.

#### 4.2. Resolution

Obtaining high resolution is also an important feature for real world PIV applications. To test the resolution of each method a testset is generated as a collection of random velocity fields, i.e. one random velocity value is generated at each pixel location (with a uniform velocity

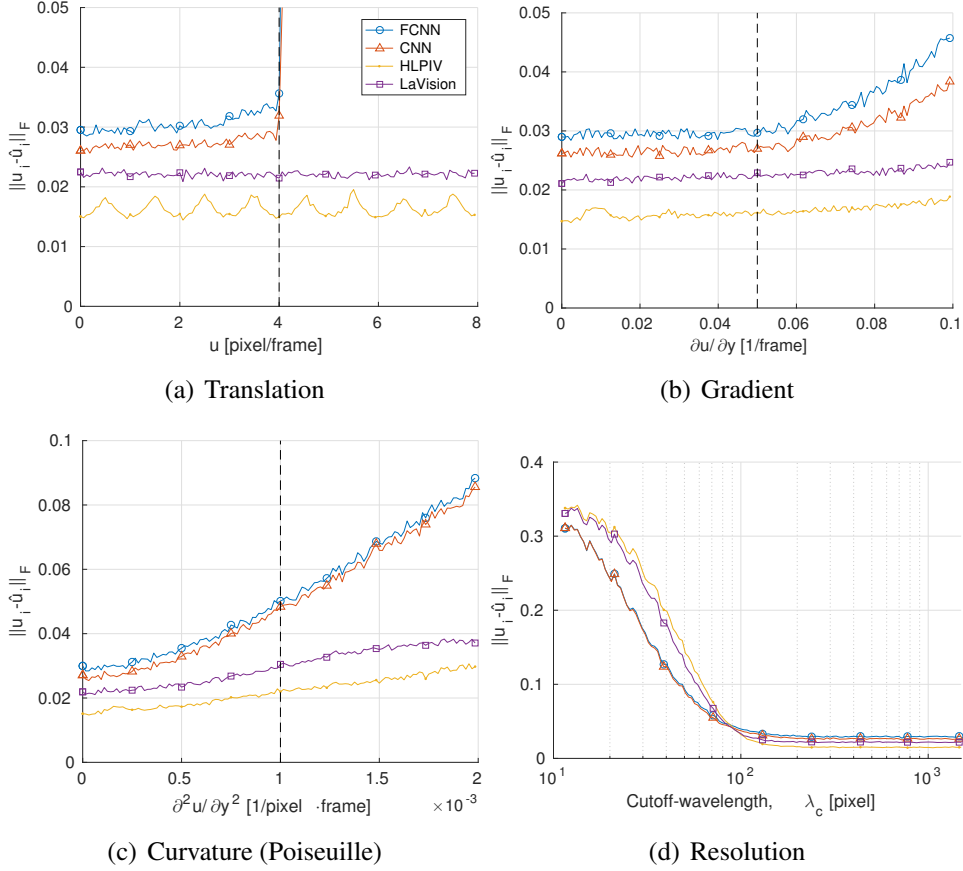


Figure 3: (a, b, c): RMS of the velocity error estimates computed at the center of the 128x128 pixels images using multipass (LaVision and HydrolabPIV), or obtained from a single 32x32 pixels window taken at the the center of the 128x128 pixels image (CNN, FCNN). The Translation, Gradient and Curvature test cases correspond to idealized velocity fields, while the last test aims at estimating the resolution obtained with each method. Vertical lines indicate the maximum value of the parameters selected when generating the pictures used for training the networks. (d): investigation of the spatial resolution of each method, performed by assessing the quality of the predictions obtained on velocity fields featuring a range of typical autocorrelation length.

spectrum, centered on zero), on which a pseudo-gaussian convolution kernel (acting as a low-pass filter) is applied. One can investigate the ability of the PIV algorithms to resolve small size structures by applying a convolution kernel with circular padding of varying radius to smooth out the local fluctuations of the velocity field on a varying length scale and estimate the RMS error of the predictions obtained. The smoothing convolution kernel is chosen as:

$$|1 - \varepsilon r|_+^4 (4\varepsilon r + 1), \quad (10)$$

where  $r = \sqrt{x^2 + y^2}$  is the radial distance from the center of the kernel,  $L = 1/\varepsilon$  is the kernel radius and  $|x|_+ = \max(x, 0)$ . The formula describing the smoothing convolution kernel is chosen so that it looks like a gaussian kernel, but with compact support. The cutoff wavelength

Table 1: Velocity error estimates for both the simplified testsets described in section 4.1 and a full validation testset, representative of the complete training set.

	$\ u_i - \hat{u}_i\ _F$ [pixel/frame]			
	translation	gradient	curvature	full
ANN	0.031 ( $\pm 0.017$ )	0.029 ( $\pm 0.017$ )	0.036 ( $\pm 0.020$ )	0.041 ( $\pm 0.024$ )
CNN	0.027 ( $\pm 0.015$ )	0.027 ( $\pm 0.015$ )	0.034 ( $\pm 0.019$ )	0.037 ( $\pm 0.022$ )
HydrolabPIV	0.016 ( $\pm 0.009$ )	0.016 ( $\pm 0.009$ )	0.018 ( $\pm 0.010$ )	0.020 ( $\pm 0.011$ )
LaVision	0.022 ( $\pm 0.013$ )	0.022 ( $\pm 0.012$ )	0.025 ( $\pm 0.014$ )	0.026 ( $\pm 0.014$ )

$\lambda_c$  is computed from the 2D FFT of the kernel, and corresponds to the scale at which 50% of the energy in the velocity spectrum has been suppressed by the low-pass filtering.

Results are presented in Fig. 3 d. The resolution of the ANNs is limited by the validity of the Taylor expansion assumed in the training sets, i.e. when the features become too small the Taylor expansion is not a good approximation for the velocity field and the training set is not representative of the test case. In addition, there are probably inherent limitations for the resolution that can be attained, for example in the CNN case due to the size of the convolution kernels. The ANNs are observed to have a better resolution than the traditional PIV codes: while traditional codes have lower error than ANNs for large  $L$  (i.e., velocity fields without sharp gradients), the opposite is observed for small  $L$  (i.e., velocity fields with sharp gradients). This is a welcome feature as conventional PIV easily underresolves complex flows [16]. The reduced resolution of traditional PIV codes is caused by using information from neighbouring subwindow in the window deformation step of the multipass process, which is avoided in the ANN case.

#### 4.3. Comparison on the full training domain

A performance comparison on an image set representative of the full training domain (but generated separately from the training set) is also used to evaluate the ANNs compared with traditional PIV softwares. Mean results, together with the ones corresponding to the part of the simplified velocity fields that lie in the training domain, are summarized in Table 1. As summarized in Table 1 both ANNs are found to have a reasonable accuracy, with the CNN doing slightly better. While not as good yet as the conventional state-of-the-art multipass PIV algorithms fed with 128x128 pixels images, results from ANNs are a lot better than what would be obtained from single pass PIV on 32x32 pixels images (which yield RMS errors of typically  $\sim 0.10 \text{ pixel/frame}$ ).

#### 4.4. Boundary performance for simplified velocity fields

Traditionally, most error analysis on PIV is done for interior velocity vectors, and this is therefore how we designed the comparisons we presented so far. However, boundary

performance is also a critical real world problem when performing PIV [40]. Performing PIV predictions near boundaries is challenging, as part of the subwindow to process is masked when predictions should be performed at a distance to the wall less than half the subwindow size. In addition, multipass methods rely on the velocity field being available in the whole neighbourhood of the point where velocity should be estimated for performing windows deformation, which is not the case either close to the boundaries. Therefore, ANNs should be less impacted by the presence of a wall.

To assess the ability of each method to perform predictions near the boundaries, the full vector fields obtained from 128x128 images processed by LaVision and HydrolabPIV were compared to the results obtained using the ANNs on 289 32x32 subwindows per image, generated using 75% overlap. This was done for 442 128x128 images giving a total of 127738 32x32 subwindows being evaluated by the ANNs. The RMS and bias errors were estimated by averaging over the number of images and along the x-axis. The vectors used in the averaging process are those for which the masking is a function of the y-position only, i.e. we do not include the corners of the images. This results in the two outermost vectors on each side being computed from subwindows masked at 50% and 25%.

Comparisons between results obtained with all PIV techniques are presented in Fig. 4. The Translation, Gradient and Curvature cases are directly inspired by the test cases presented in section 4.2. As explained previously for each test case 442 image realizations, that feature the same velocity field but different random particle images locations and sizes, are used to generate the statistics presented. The Translation field is simply a purely translational field, with a displacement of 2 pixels/frame. The Gradient field is constant along the X direction and presents a constant gradient in the Y direction which is chosen so that the maximum displacements at the top and bottom of the field are -4 and +4 pixels/frame, respectively. The resulting gradient value is -0.0625 per frame, which is slightly outside of the training range of the ANNs. The Poiseuille field is chosen so that there is zero displacement at the top and bottom of the field, and a 2 pixels/frame maximum displacement at the center of the field. The resulting value of curvature is  $0.92 \cdot 10^{-4}$  frame/pixel, which is close to the maximum value for which the ANNs were trained.

Fig. 4 a and b summarize the results obtained with the Translation test image. The rms error (which includes both variance and bias errors) is small for all methods, with a slight advantage for conventional PIV which is expected as these algorithms are built on the assumption that the displacement is approximately linear. None of the methods have any significant bias error.

Fig. 4 c and d summarize the results obtained with the Gradient test image. As visible in the figure c, LaVision makes a slight simplification when evaluating the normalized cross-correlation, which results in boundary artifacts mostly contributing to the bias error. This can be improved by using (more computationally expensive) masked normalized cross-correlation [34], as done in HydrolabPIV. As a result boundary effects are reduced in HydrolabPIV, though not completely eliminated either. ANNs, that do not use information from adjacent subwindows, have good boundary accuracy except at the point exactly on the boundary for which 50 percent of the image is missing. However it should be possible to train ANNs with

Table 2: Comparison of the computational times (in seconds) for 128000 vectors measured using CNN, FCNN and LaVision. HydrolabPIV is not included as its aim is to have an easy to modify code for testing PIV algorithms, and as a consequence computation time is not a priority.

	CNN		FCNN		LaVision	
	GTX970	GTX980TI	GTX970	GTX980TI	CPU	GTX970
Loading to RAM from HDD	8.2	5.4	7.5	6.0		
Generating slice from RAM	0.4	0.3	0.2	0.2		
Computation on GPU	22.9	7.2	15.9	5.1	-	-
Writing result to HDD	1.5	0.9	1.5	1.0		
Total time	33.0	13.8	25.1	12.3	70.6	55.0

masking, which would very likely improve further their boundary accuracy in this case.

Fig 4 e and f summarize the results obtained with the Curvature test image. ANNs display more important bias error in the center of the image than other PIV algorithms, which was discussed previously in section 4.1 as being probably a consequence of the Hessian not being used in the training of the ANNs. As for the Gradient test, additional bias errors due to the boundaries are small and ANNs perform best of all models close to the image boundaries, with a good margin over LaVision.

#### 4.5. Computation speed

Processing streams of high frequency, high resolution images creates a significant computational burden, and computation speed is one of the limits encountered when performing PIV. While innovative techniques, such as the use of FGPA, have been presented for addressing this issue [23, 24], resorting to CPUs and GPUs remains the norm. The computation time needed for performing predictions on 128000 image pairs with each ANN architecture using two different GPU models is compared with the time needed by LaVision, which is used following the method indicated in section 3.1. HydrolabPIV is not included as its aim is to have an easy to modify code for testing PIV algorithms, and as a consequence computation time is not a priority, making it significantly slower than LaVision. The test case used corresponds to the pure Curvature test (Poiseuille) presented in section 4.4. Results are presented in Table 2.

While LaVision is claimed to attain speedup in the range of x10 for Stereographic and 3D PIV [4], a much more modest speedup was observed in our 2D PIV case. This may be because more effort was put in developing fast 3D PIV on GPUs than for 2D PIV. Using LaVision, no speedup was obtained using a GTX980TI compared with a GTX970. ANNs perform faster than LaVision and take full advantage of the performance gain between GTX970 and GTX980TI GPUs, with speedups relative to LaVision up to x4. As can be seen in Table 2, the time needed to read the images data from HDD is comparable with the time needed to perform computations on the most powerful GPU. While this implies that one

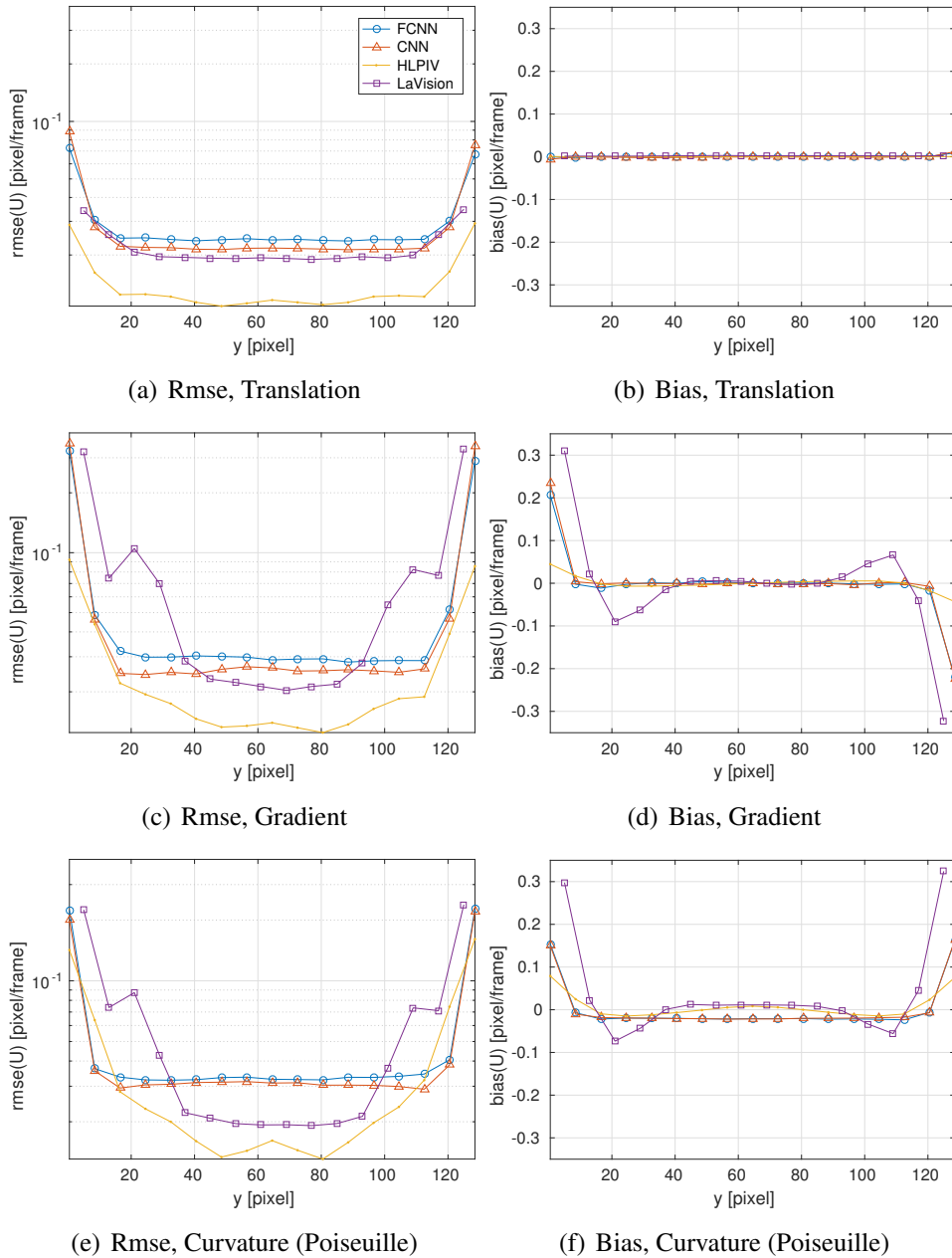


Figure 4: RMS error (left) and bias error (right) of the X component of the velocity estimates obtained with each PIV algorithm on a series of 442 128x128 pixels test image pairs. The results shown for LaVision and HydrolabPIV are extracted from the last of the PIV pass. The results shown for both the CNN and the FCNN are obtained from feeding them with a series of 32x32 pixels subwindows sampled with a 75% overlap. All results presented are averaged over the 442 image realizations and along the x-axis of the images, see the text for more details. From top to bottom the 128x128 pixels images used correspond to a pure translation, pure gradient and pure curvature velocity field.

should be careful with minimizing readings from HDD, this should not be a limitation for real world applications when large images are processed and overlap is used.

Performing fair benchmarking of codes can be challenging. In a real world application, some of the computations performed by LaVision will be re-used for evaluating neighboring velocity estimates on images bigger than 128x128 pixels. However, the same applies to at least CNNs. The results of the convolution step, which is both time and memory expansive, could be shared between adjacent velocity calculations when overlap is used. Therefore, the global picture given by our benchmarks should hold also for the PIV analysis of complete images, and the CNN computation time could even be further reduced by implementing sharing of the convolution outputs between adjacent subwindows.

#### 4.6. Test on real world data

The CNN, which is the best-performing of the two ANNs based on the results presented so far in this section, is tested on some real world data. Due to the limitations of the proof-of-concept ANNs that were trained, we must limit ourselves to images with a range of particle images displacement of  $[-4, 4]$  pixels in both the x and y directions. Some images fulfilling this criterion were recently recorded by the first author of this paper during another project, and are therefore used here. The images recorded correspond to exponentially damped water waves propagating under a slush ice layer. A Falcon2 4M camera was used to record images at a rate of 75 frames per second. A high power water-cooled LED array was used to provide illumination, and 50  $\mu\text{m}$  spherical Polyamid Seeding Particles (PSP) were used as tracers. In the present study, we use the first two images of a run for performing the benchmarking.

The first step in using the CNN on real world data consists in normalising the images so that they present similar properties as the training set. If one uses the raw images without performing such normalisation, the vector field generated by the CNN is noisy and many outlier vectors are present. There are two main properties of the real world data that need to be renormalised to successfully use the CNN. Firstly, the real world images are saturated on several pixels near the center of each particle image, due to excessive light intensity used during image acquisition. To reduce this effect, the raw images are convolved with a kernel of size  $3 \times 3$  with small coefficients outside of the center of the kernel, so that the images get smoother and the saturation effect is reduced. Secondly, the histogram of the pixels intensities is renormalised so that it is approximately equal to the one of the training set. This is performed by binning the pixels intensities and changing the values of the bins, while preserving their ordering, so that the difference between the training and the real world histograms is minimized. The renormalisation is performed on each  $32 \times 32$  sub-window, before being fed into the network.

Results for the  $u$  and  $v$  velocity components obtained with the CNN, HydrolabPIV and LaVision were examined by the authors. HydrolabPIV was the best performing cross-correlation software from the tests performed on synthetic data, therefore its results together with the ones from the CNN are presented in Fig. 5. LaVision was also used to analyse the images, and results similar to HydrolabPIV and the CNN were found (not reproduced



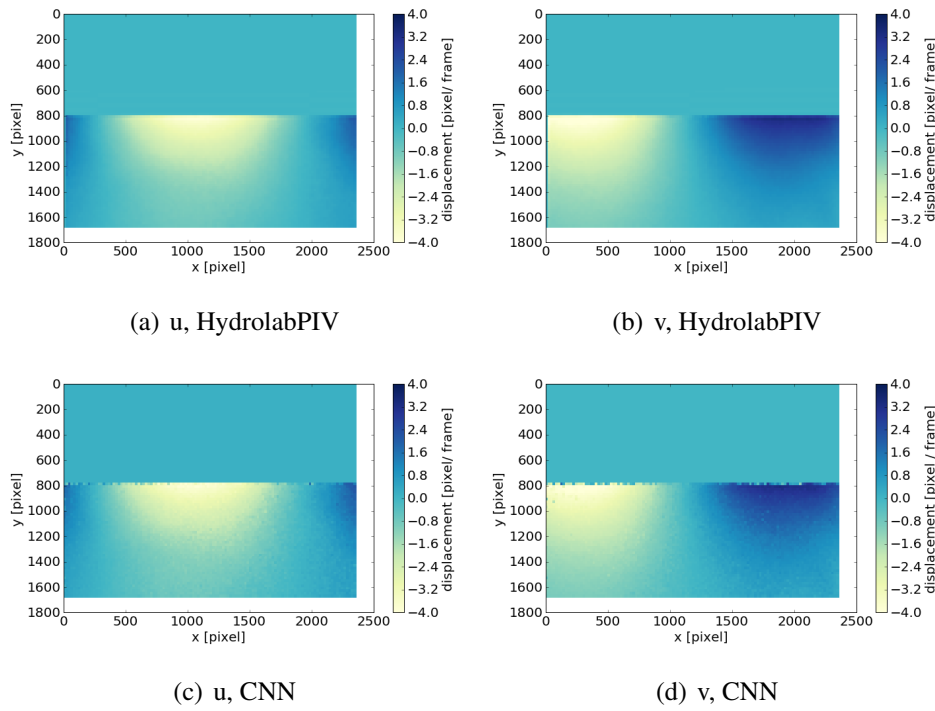


Figure 5:  $u$  and  $v$  velocity components obtained using HydrolabPIV and the CNN, from real-world images. The results of both methods are very similar. Slightly higher noise levels are observed with the CNN, which could be reduced using an outlier detection and interpolation similarly to the built-in functionality of HydrolabPIV. One line of disturbed results is visible in the CNN output, as all subwindows are processed, including those for which 75 % or more of the subwindow is masked. LaVision was also used to analyse the images, but the results are not presented here as they are very similar.

here). As the mechanisms for selecting sub-windows is slightly different between the two programs, the output velocity fields from the CNN are translated of around half a subwindow and both velocity fields from the CNN and HydrolabPIV are interpolated on a common grid for performing the comparison. As visible in Fig. 5, both methods produce very similar results. The output of the CNN is slightly more noisy than the output of HydrolabPIV, which could be improved by using an outlier detection filtering on the output from the CNN, similarly to the built-in functionality of HydrolabPIV. One line of vectors close to the surface is slightly distorted in the case of the CNN, which is due to the fact that all subwindows are computed by the CNN, including those that feature a masked region representing 75 % or more of the subwindow.

Both the error maps of the horizontal component of the velocity, computed as the difference between the  $x$  velocity component obtained from HydrolabPIV or LaVision and the CNN, and the statistical distribution of those error maps were investigated by the authors. Results corresponding to the comparison with HydrolabPIV are presented in Fig. 5. No systematic bias in the prediction of the  $x$  velocity component between the two methods is found, and the distribution of the values of the error maps is concentrated around zero

and does not present secondary peaks. The mean value and standard deviation of the error map distributions are 0.003 and 0.058  $pxls/frame$ , respectively, between Hydrolab PIV and the CNN, and  $-0.0026$  and 0.035  $pxls/frame$ , respectively, between Hydrolab PIV and LaVision. The standard deviation of the discrepancy relative to the predictions of HydrolabPIV is of the order of 1.5 % of the 4  $pxls/frame$  prediction range of the CNN, which is consistent with the results presented based on synthetic images. This comparison proves that the CNN can be used not only with synthetic images but also with more noisy, imperfect images obtained in a real experimental setup.

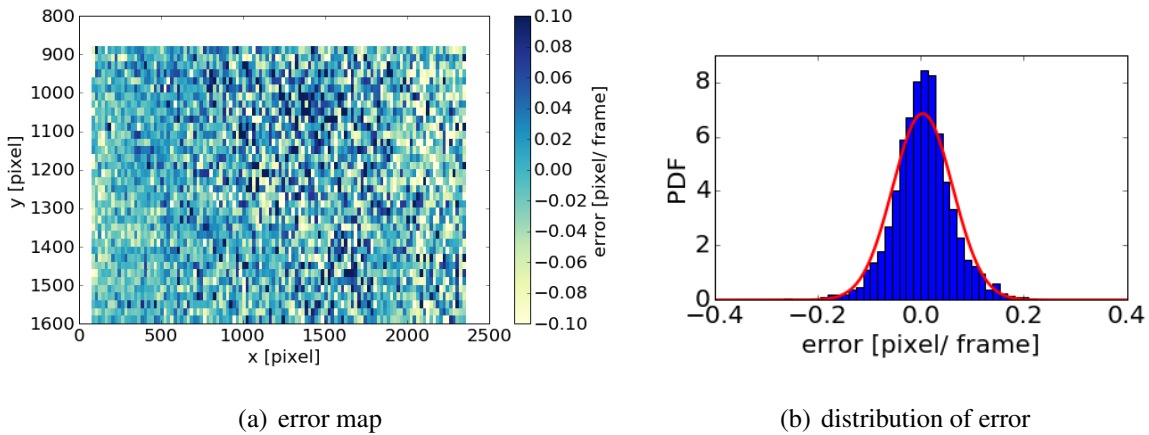


Figure 6: Error map for the x velocity component, and distribution of the error map, corresponding to the comparison between HydrolabPIV and the CNN. The mean value and standard deviation of the error map distribution are 0.003 and 0.058  $pxls/frame$ , respectively. The kurtosis of the distribution of the error map is 2.1. The best fit Gaussian curve is plotted on top of the distribution of the error map, which shows that the error distribution is more concentrated around 0 than would be a Normal law.

## 5. Conclusion

Both a Convolutional Neural Network (CNN) and a Fully Connected Neural Network (FCNN) are trained to perform end-to-end PIV on realistic synthetic data and tested on several test sets, both synthetic and from real-world data. The experimental results we report are a proof of concept of the use of Artificial Neural Networks (ANNs) for performing end-to-end PIV. This is the first time to the authors knowledge that ANNs are used to perform end-to-end PIV. The level of Root Mean Square (RMS) error between ANNs predictions and the velocity values used for generating the images is slightly higher than for state-of-the-art PIV codes, but much better than what would be expected from single pass PIV. This is a good result considering that current state-of-the-art PIV codes are the results of over 25 years of continuous improvements of the processing algorithms, while the proof of concept ANNs we present here are fairly simple in their design and better variants could very likely be developed with additional work.

Moreover, our results suggest that ANNs may have several advantages over more traditional 2D PIV methods. Benchmarking shows that ANNs are better at using efficiently GPUs than the GPU module from LaVision we had access to, which may be of interest as GPUs are seen as promising for large heavy computations. We were also able to obtain better resolution with ANNs compared with traditional PIV methods, which could be of interest in cases when high local flow variations are expected. Finally, we observed good boundary performance of ANNs compared with more traditional PIV methods. In addition, ANNs could be further trained with datasets including masked images, which should improve the network predictions near the fluid boundaries.

The present work could be extended in several ways. More sophisticated synthetic images could be used, to quantitatively take into account for example out of plane motion. Images featuring high shear or even discontinuous flows could be used during training, with the aim of overcoming the smoothing gradient effect observed with traditional cross-correlation PIV techniques. Thanks to the use of small convolution kernel size and a fully connected leaky ReLU network under the convolution layer that should be able to recognize such jumps, a reduction in smoothing parasitic effects can be expected compared with traditional cross-correlation methods. The flexibility allowed by using the leaky ReLU fully connected layer could also be expected to help solving more sophisticated problems for which it is challenging to provide a satisfactory method using a traditional approach, such as for example cases when both a phase composed of large particles in suspension and a liquid phase must be separately analysed on the same pictures. Finally the networks used are simple, and more sophisticated or recent designs such as Recurrent Neural Networks and Residual Neural Networks could be investigated. More generally, refinement of ANNs including training on a wider range of pixel displacements and thorough testing on both synthetic and real world data could make them a credible alternative to traditional PIV methods.

## 6. Acknowledgements

This work was financed by the research project DOMT - Developments in Optical Measurement Technologies (project number 231491) and the PETROMAKS2 233901 project, both funded by the Research Council of Norway.

## 7. References

- [1] Adrian, R.J.: Particle-imaging techniques for experimental fluid mechanics. *Annu. Rev. Fluid Mech.* **23**, 261–304 (1991)
- [2] Dalziel, S.: DigiFlow user guide. Dalziel research partners, 3.4 edn. (2012). <http://www.dalzielresearch.com/digiflow/digiflow.pdf>
- [3] Dalziel, S.B.: Decay of rotating turbulence: some particle tracking experiments. *Applied Scientific Research* **49**(3), 217–244 (1992). DOI 10.1007/BF00384624. URL <http://dx.doi.org/10.1007/BF00384624>
- [4] DAVIS: DAVIS Software for Intelligent Imaging. Brochure (accessed 04/2017). URL <http://www.lavision.de/en/downloads/brochures.php>
- [5] Drazen, D., Lichtsteiner, P., Häfliger, P., Delbrück, T., Jensen, A.: Toward real-time particle tracking using

- an event-based dynamic vision sensor. *Experiments in Fluids* **51**(5), 1465 (2011). DOI 10.1007/s00348-011-1207-y. URL <http://dx.doi.org/10.1007/s00348-011-1207-y>
- [6] Ghosal, S., Mehrotra, R.: Robust optical flow estimation using semi-invariant local features. *Pattern recognition* **30**(2), 229–237 (1997)
- [7] Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10)*. Society for Artificial Intelligence and Statistics (2010)
- [8] Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: G.J. Gordon, D.B. Dunson (eds.) *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11)*, vol. 15, pp. 315–323. *Journal of Machine Learning Research - Workshop and Conference Proceedings* (2011). URL <http://www.jmlr.org/proceedings/papers/v15/glorot11a/glorot11a.pdf>
- [9] Goodfellow, I., Bengio, Y., Courville, A.: *Deep learning* (2016). URL <http://www.deeplearningbook.org>. Book in preparation for MIT Press
- [10] Grant, I., Pan, X.: An investigation of the performance of multi layer, neural networks applied to the analysis of PIV images. *Experiments in Fluids* **19**(3), 159–166 (1995). DOI 10.1007/BF00189704. URL <http://dx.doi.org/10.1007/BF00189704>
- [11] Grant, I., Pan, X.: The use of neural techniques in PIV and PTV. *Measurement Science and Technology* **8**(12), 1399 (1997). URL <http://stacks.iop.org/0957-0233/8/i=12/a=004>
- [12] Gui, L., Merzkirch, W.: A comparative study of the mqd method and several correlation-based piv evaluation algorithms. *Experiments in Fluids* **28**(1), 36–44 (2000). DOI 10.1007/s003480050005. URL <http://dx.doi.org/10.1007/s003480050005>
- [13] Gui, L.C., Merzkirch, W.: A method of tracking ensembles of particle images. *Experiments in Fluids* **21**(6), 465–468 (1996). DOI 10.1007/BF00189049. URL <http://dx.doi.org/10.1007/BF00189049>
- [14] Hassan, A.Y., Philip, G.O.: A new artificial neural network tracking technique for particle image velocimetry. *Experiments in Fluids* **23**(2), 145–154 (1997). DOI 10.1007/s003480050096. URL <http://dx.doi.org/10.1007/s003480050096>
- [15] Haykin, S.: *Neural Networks: A Comprehensive Foundation*, 2nd edn. Prentice Hall PTR, Upper Saddle River, NJ, USA (1998)
- [16] Herpin, S., Wong, C.Y., Stanislas, M., Soria, J.: Stereoscopic piv-measurements of a turbulent boundary layer with large spatial dynamic range. *Experiments in Fluids* **45**, 745–763 (2008)
- [17] Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. *Neural Networks* **2**(5), 359 – 366 (1989). DOI [http://dx.doi.org/10.1016/0893-6080\(89\)90020-8](http://dx.doi.org/10.1016/0893-6080(89)90020-8). URL <http://www.sciencedirect.com/science/article/pii/0893608089900208>
- [18] Huang, H., Fiedler, H., Wang, J.: Limitation and improvement of PIV, part I. *Experiments in Fluids* **15**, 168–174 (1993)
- [19] Huang, H.T., Fiedler, H.E., Wang, J.J.: Limitation and improvement of PIV, part II. *Experiments in Fluids* **15**(4), 263–273 (1993). DOI 10.1007/BF00223404. URL <http://dx.doi.org/10.1007/BF00223404>
- [20] Keane, R.D., Adrian, R.J.a.: Super-resolution particle image velocimetry. *Measurement Science and Technology* **6**, 754–768 (1995)
- [21] Kolaas, J.: Optimization of optical measurement techniques in fluid mechanics with application to micro fluidics, multiphase flow and water waves. Ph.D. thesis, University of Oslo (2014)
- [22] Kolaas, J.: Getting started with HydrolabPIV v1.0. Preprint series. Research Report in Mechanics <http://urn.nb.no/URN:NBN:no-53997> (2016)
- [23] Kreizer, M., Liberzon, A.: Three-dimensional particle tracking method using fpga-based real-time image processing and four-view image splitter. *Experiments in Fluids* **50**(3), 613–620 (2011). DOI 10.1007/s00348-010-0964-3. URL <http://dx.doi.org/10.1007/s00348-010-0964-3>
- [24] Kreizer, M., Ratner, D., Liberzon, A.: Real-time image processing for particle tracking velocimetry. *Experiments in Fluids* **48**(1), 105–110 (2010). DOI 10.1007/s00348-009-0715-5. URL <http://dx.doi.org/10.1007/s00348-009-0715-5>
- [25] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: F. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (eds.) *Advances in*

- Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012). URL <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>
- [26] Kutz, J.N.: Deep learning in fluid dynamics. *Journal of Fluid Mechanics* **814**, 14 (2017). DOI 10.1017/jfm.2016.803
- [27] Labonté, G.: A new neural network for particle-tracking velocimetry. *Experiments in Fluids* **26**(4), 340–346 (1999). DOI 10.1007/s003480050297. URL <http://dx.doi.org/10.1007/s003480050297>
- [28] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**, 436–444 (2015). DOI <http://dx.doi.org/10.1038/nature14539>. URL <http://dx.doi.org/10.1038/nature14539>
- [29] LeCun, Y., Boser, B., Denker, J.S., Howard, R.E., Hubbard, W., Jackel, L.D., Henderson, D.: Advances in neural information processing systems 2. chap. Handwritten Digit Recognition with a Back-propagation Network, pp. 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1990). URL <http://dl.acm.org/citation.cfm?id=109230.109279>
- [30] LeCun, Y., Bottou, L., Orr, G., Muller, K.: Efficient backprop. In: G. Orr, M. K. (eds.) *Neural Networks: Tricks of the trade*. Springer (1998)
- [31] Liang, D., Jiang, C., Li, Y.: Cellular neural network to detect spurious vectors in PIV data. *Experiments in Fluids* **34**(1), 52–62 (2003). DOI 10.1007/s00348-002-0530-8. URL <http://dx.doi.org/10.1007/s00348-002-0530-8>
- [32] Merzkirch, W., Gui, L., Hilgers, S., Lindken, R., Wagner, T.: PIV in multiphase flow. In: *The second international workshop on PIV*, pp. 8–11 (1997)
- [33] Nobach, H., Honkanen, M.: Two-dimensional Gaussian regression for sub-pixel displacement estimation in particle image velocimetry or particle position estimation in particle tracking velocimetry. *Experiments in Fluids* **38**(4), 511–515 (2005). DOI 10.1007/s00348-005-0942-3. URL <http://dx.doi.org/10.1007/s00348-005-0942-3>
- [34] Padfield, D.: Masked object registration in the fourier domain. *IEEE Transactions on image processing* **21**(5), 2706–2718 (2012)
- [35] Quénot, G.M., Pakleza, J., Kowalewski, T.A.: Particle image velocimetry using optical flow for image analysis. In: *8th Int. Symposium on Flow Visualization*, pp. 47–1 (1998)
- [36] Raffel, M., Willert, C.E., Wereley, S.T., Kompenhans, J.: *Particle image velocimetry : a practical guide*. Experimental fluid mechanics. Springer, Berlin, New York (2007). URL <http://opac.inria.fr/record=b1125276>
- [37] Rosenblatt, F.: The perceptron—a perceiving and recognizing automaton. Tech. Rep. 85-460-1, Cornell Aeronautical Laboratory (1957)
- [38] Ruhnau, P., Kohlberger, T., Schnörr, C., Nobach, H.: Variational optical flow estimation for particle image velocimetry. *Experiments in Fluids* **38**(1), 21–32 (2005)
- [39] Siegelmann, H., Sontag, E.: On the computational power of neural nets. *Journal of Computer and System Sciences* **50**(1), 132 – 150 (1995). DOI <http://dx.doi.org/10.1006/jcss.1995.1013>. URL <http://www.sciencedirect.com/science/article/pii/S0022000085710136>
- [40] Theunissen, R., Scarano, F., Riethmuller, M.L.: On improvement of piv image interrogation near stationary interfaces. *Experiments in Fluids* **45**, 557–572 (2008)
- [41] Uhrig, R.E.: Introduction to artificial neural networks. In: *Industrial Electronics, Control, and Instrumentation, 1995.*, Proceedings of the 1995 IEEE IECON 21st International Conference on, vol. 1, pp. 33–37 vol.1 (1995). DOI 10.1109/IECON.1995.483329
- [42] Willert, C.E., Gharib, M.: Digital particle image velocimetry. *Experiments in Fluids* **10**(4), 181–193 (1991). DOI 10.1007/BF00190388. URL <http://dx.doi.org/10.1007/BF00190388>
- [43] Zitova, B., Flusser, J.: Image registration methods: a survey. *Image and vision computing* **21**(11), 977–1000 (2003)