Using deep learning to retrieve 3D geometrical characteristics of a particle field from 2D projected images: Application to multiphase flows

Kassem Dia, Fabrice Lamadie, Johan Debayle

Using deep learning to retrieve 3D geometrical characteristics of a particle field from 2D projected images: Application to multiphase flows

Kassem Dia
CEA, DES, ISEC, DMRC
Univ Montpellier
Marcoule, France
kassem.dia@cea.fr

Fabrice Lamadie
CEA, DES, ISEC, DMRC
Univ Montpellier
Marcoule, France
fabrice.lamadie@cea.fr

Johan Debayle
MINES Saint-Etienne,
SPIN/LGF UMR CNRS 5307
158 cours Fauriel, Saint-Etienne, France
debayle@emse.fr

***Abstract*—The main part of recycling processes are carried out in chemical engineering reactors that involve multiphase flows with dense dispersed phase. In a study and modeling approach of these processes, the description and characterization of hydrodynamic phenomena is crucial. A variety of techniques allows us to realize this type of measurement, but the most used one is the direct imaging associated with an efficient image processing. Recently, deep learning algorithms have proven to be very effective in solving image based problems, which led to the use of these algorithms to extract critical information in chemical engineering apprentices. The method employed in this paper relies on a deep learning based algorithm dedicated to the prediction of 3D features of multiphase flows using 2D projected images. The performance of the method has been evaluated both on synthetic images and on real images of beads in a dispersed phase.**

***Index Terms*—deep learning, multiphase flow, 3D stochastic modeling**

I. Introduction

Most of the R&D studies on recycling processes (critical metals, rare earths, etc.) are currently carried out on the basis of laboratory-scale experiments and modeling. As the unit steps of these processes (e.g. dissolution, liquid-liquid extraction, precipitation, filtration, leaching, etc.) take place in a special equipment (pulsed columns, mixer settlers,etc.), the description of multiphase flows in these devices is a key issue, especially for scaling up. As a consequence, in order to complement the chemical investigations, studying the hydrodynamics of these phenomenons is one way to apprehend and predict the effect of the flow on the efficiency of these chemical processes. In that event, identifying the flow's key properties in the related apparatus, specially the geometrical properties of the dispersed phase (e.g. void fraction, particles size and shape distribution, etc.) is required.

Multiple approaches have already been employed by different researchers in order to retrieve these information. One of the most used methods consists in using image processing techniques on 2D images, as done by Hosokawa *et al.* [Hosokawa et al., 2009] Riquelme *et al.* [Riquelme et al., 2013], Khalil *et al.* [Khalil et al., 2010], Bian *et al.* [Bian et al., 2013] using the Hough transform to detect and measure the flow's properties, or the watershed transformation used by Lau *et al.* [Lau et al., 2013] to retrieve the bubble size distribution. On the other hand, the most advanced algorithms relied on concavity points detection in particle's clusters like the work done by Zhang *et al.* [Zhang et al., 2012], Farhan *et al.* [Farhan et al., 2013], Langlard *et al.* [Langlard et al., 2018a], in order to separate overlapping objects and measure the flow's properties. Effectively, the use of these approaches have two major drawbacks: 1) In terms of accuracy, the algorithms are rendered useless in the presence of a huge number of overlapped particles in the images, and 2) they only deal with 2D characteristics of the projected images and where unable to establish a direct link between 2D and 3D characteristics.

In the past decades, deep learning took a huge step into the image processing field. Due to the up-rise and evolution of these algorithms, researchers became reliant on these techniques now more than ever, this includes the image based problems in the chemical engineering field. Most of the work present in the literature relied on a specific deep learning algorithm called Convolutional Neural Network (CNN), such as the work done by Kim *et al.* [Kim and Park, 2021], and Haas *et al.* [Haas et al., 2020]. They have used these algorithms to detect and extract bubbles in complex multiphase flows 2D images. Others used a different kind of CNN that relies also on convolutional operators but with an architecture that takes a U shape. It is called the U-Net. Li *et al.* [Li et al., 2020], Seredkin *et al.* [Seredkin et al., 2020], Ronneberger *et al.* [Ronneberger et al., 2015] used U-Net to detect overlapping bubbles in 2D images. This type of approach has proven to be rather efficient but has the same drawback as classical image processing techniques, i.e. that the information retrieved only concerns the 2D features.

In this contribution, a deep learning method using CNN algorithms is proposed to retrieve 3D characteristics from 2D projected images of multiphase flows, more precisely the density of the particles and their size distribution in a continuous phase. It consists in training a CNN network

using 2D projections of the 3D stochastic model used by de Langlard *et al.* [Langlard et al., 2018b]. The resulting CNN network is then tested and validated on real 2D images of beads in a continuous phase. Note that in this study we only focus on studying images that contains strictly spherical shaped particles.

II. Methodology

A. Motivation

Over the past decade, deep learning algorithms, including convolutional neural networks, have been increasingly used due to their efficiency and simplicity compared to conventional methods. Unfortunately, a CNN network requires a large amount of labeled data set in order to be trained correctly. In our case, it is nearly impossible to access, precisely and in a large quantity, some of these critical information of the flow. Therefore, modeled images (also called *synthetic images* in this paper) of multiphase flows could resolve this problem, where the ground truth information is easily accessible and can be generated in large quantities. In order to generate this type of data we refer to stochastic geometry models, where they showed their efficiency to characterize a particle field in a multiphase flow [Langlard et al., 2018b].

B. 3D image characterization using stochastic geometry-based approaches

In order to generate synthetic images to train a CNN network, we will refer to the 3D stochastic geometrical model used by de Langlard *et al.* [Langlard et al., 2018b] to generate a 3D field of spherical particles in a continuous phase. It is a generalization of a Matérn type II point process, which itself is a thinning of an underlying homogeneous Poisson point process in $\mathbb{R}^d$ of intensity λ. The thinning rule consist of introducing a hard-core distance, denoted by R, between the generated points, where it eliminates the last arriving points that are closer than $2R$. Such a process can be considered as a marked Poisson point process, where the first mark is R (constant and positive) and the second mark stands for the time of arrival. The generated 3D field is orthogonally projected into a 2D image to create the synthetic images (Figure 1).

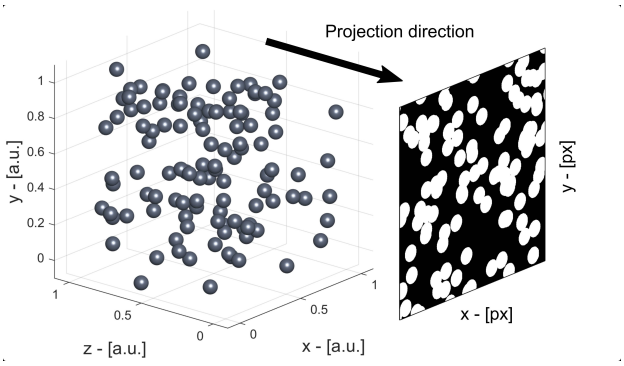

Fig. 1. A 3D illustration of the modeling approach. A 3D geometrical model of hardcore particles is constructed and orthogonally projected in order to construct the 2D images.

C. Convolutional neural networks

1) Description: A convolutional neural network is a supervised deep learning algorithms, meaning that each data point is labeled with a representative information, where the network should learn to predict these labels. These algorithms are mainly used to process data with a grid-like topology (e.g. images, financial time series, etc.). And more recently, they have become more popular for image analysis problems because of their efficiency compared to conventional techniques.

Usually, CNN architectures are composed of four different parts used to build the network. The ***input layer*** that hold the information of the input data (e.g. the pixel values of an image). ***Convolutional layers*** that automatically capture the features in each image (e.g. lines, shapes, specific objects, etc.). These layers contain a set of filters (e.g. Kernels) whose parameters are optimized during the training process. Each one of these filters convolves with the input volume creating an activation map (also called a feature map) that is stacked one over the other. Finally, each of the resulting volume subjected to a non-linear activation function (e.g. ReLU, sigmoid, etc.) to increase the non-linearity of the network. Then, ***pooling layers*** are used to reduce the dimensions of the feature maps, thus saving computational time, but more importantly removing all kinds of unwanted noise and distortions that might be present in the feature map and affect the accuracy of the training. Finally, ***fully-connected layers*** is placed at the end of the network containing multiple neurones. The input to these layers is a reshaping of the resulting feature maps into a vector. All the components of the input vector are connected to all neurons of a hidden layer. The values of these neurons are computed by multiplying the input vector by a weight that is iteratively optimized during the training process. The final hidden layer is used to output the final prediction. It can consist of one or more nodes, and it can represent either classes in a classification problem or specific quantities in a regression problem.

2) Training process: During the training process, the hyperparameters of the network are tuned by forward-backward propagation. The data set used to train the network is composed to two different sets, a training data set that is used to generalize and adapt a suitable network capable of solving the problem at hand. The second set is called a validation data set, that provides an unbiased evaluation of the network's performance on the training data set while tuning its hyperparameters. A loss function measures the overall performance of the network. This gradient-based optimization reduce the overall loss, and the procedure is repeated multiple times for all training images, slowly adjusting the weights of the fully connected layers. One processing of all images of the training data set is called an epoch. The model usually needs multiple epochs in order to learn correctly.

III. Network training

A. Building the network

There is no theoretical method to build the optimal convolutional network, but rather some basic principles should be

followed that optimize the overall performance and make the network more reliable in performing specific tasks. The idea is to start with a simple architecture that is capable of detecting low-level features. This type of network is perfect in some cases where the features are easy to detect and sufficiently representative of the data point. But this is not always the case, where sometimes the use of a simple architecture to collect hard information will cause the model to *underfit*, meaning that the model is unable to capture a relationship between the input data and their labels. In this case, the number of convolutional layers should be increased in order to detect high-level features, thus helping in learning and generalizing better over the global structures of the input data (here the input is an image with three channels). On the other hand, this process of layer adding can also be harmful for the network, where in some cases the network is trained to search for some overly complex features that has no connection to the labels. This is the case of *overfitting*, where some non-representative information will be captured in the images by the convolution layers, thus generalizing a network over a particular set of data (e.g. the training set), and failing to reliably predict any future observations. Therefore, during the training process, one should evaluate the performance of the network over multiple epochs by monitoring the learning curves. The fit is at its best if the training and validation losses decrease to a point of stability and if the gap between the two curves is minimal.

B. The network description

Here we present the structure of the four different CNNs that were employed in this study. The images used to train these architectures are 2D projections of a 3D particle field (Figure 1). Moreover, each CNN is assigned to predict a specific 3D characteristic. The tasks are assigned as follows.

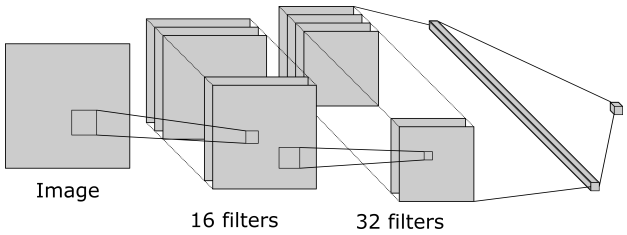

Fig. 2. A diagram representing the CNN^{VF} architecture.

- **CNN^{VF}:** The goal of this network is to determine the the volume fraction of spherical particles in a 3D field, VF_{3D}, from the 2D projected images of a monodispersed spherical particles. The architecture of this network is composed of two convolutional layers, they contain 16 and 32 filters respectively with a size of 3×3, in addition to a ReLU activation function. After each convolutional layer, a 2×2 max pooling layer is applied. The fully connected layer is a flattening vector of the feature maps with a dropout of 50%, which means that half of the neurons are ignored during the training phase. This procedure reduces overfitting. Finally, a single output is attached to the network indicating the predicted value, $\hat{VF}_{3D}$. Check figure 2.
- **CNN^{R}:** The second network is trained to determine the radius of the particles, R_{3D}, in a monodispersed spherical particles, where the particles size varies from one simulation to another with a fixed particle count. After we tested the architecture of CNN^{VF}, the network did not perform as well as it should. Therefore, two more convolutional layers were added containing 64 and 128 filters respectively, each followed by 2×2 max pooling layer, in addition to a full connected layer containing 32 neurons after the dropout. The output of this network is a continuous value, therefore it is rounded to the nearest integer which is considered as the predicted radius, $\hat{R}_{3D}$.
- **CNN^{R2}:** The third network is dedicated to predicting two outputs: the small radius and the large radius of a bidispersed spherical particles. Therefore, a fifth convolutional layer of 128 filters and a max pooling layer are added, and the fully connected layer of 32 neurons is replaced by one containing 64 neurons, and the output is two neurons representing the small $\hat{R}1_{3D}$ and the large radius $\hat{R}2_{3D}$. Same as the previous CNN, the values are rounded.
- **CNN$^{\mu}$:** The fourth, and final network, will have the same architecture as the CNN^{R2} but with a single output. The goal of this network is to estimate the mean radius, μ_{3D}, in a multi dispersed phase, where the radii varies from one particle to another in a single image. The output is a continuous value indicating the mean value of the radii.

All off these architectures were determined by an iterative approach based on several dozen tests.

IV. Results and Discussions

Each one of these networks is trained and validated using 16000 and 2000 labeled synthetic images, respectively, produced by the 3D stochastic model. The training is done over 300 epochs and a learning rate is fixed at 0.0001 (i.e. optimization speed) with a batch size of 32 images per iteration (i.e. the number of samples to work through before updating the internal model parameters). The loss function used to evaluate the training process is called the mean squared error (mse) which computes the mean distance between the ground truth and the predicted values. On the other hand, the absolute percentage error (ape) is used to measure the error percentage for each prediction.

$$ape_i = \frac{|\hat{y}_i - y_i|}{y_i} \times 100 \quad \forall i \in \mathbb{N}^* \tag{1}$$

where $\hat{y}_i$ is the predicted value and y_i is the ground truth. Therefore, ape is used to color code the predictions as follows:

- : ape $\leq 5\%$
- : $5\% <$ ape $\leq 10\%$
- : $10\% <$ ape

A. Estimating the volume fraction

In this section we will focus on estimating the volume fraction of the dispersed phase in a continuous phase. The objective here is to train the first CNN network, CNN^{VF}, capable of predicting the volume fraction in a monodispersed spherical particles flow, where all the particles are of the same size but their count differs from one simulation to another. The images used to train the network contains particles with a radius of 61 pixels (apparent radius of the beads through the optical system used in the experiments) with a volume fraction of the dispersed phase that varies from $0.2\%_{VF}$ to $8\%_{VF}$. The accuracy of the model is tested over 2000 synthetic images of the same law. The results are presented in figure 3. 75% of the predictions had an error rate less than 10% which is totally suitable for our application.

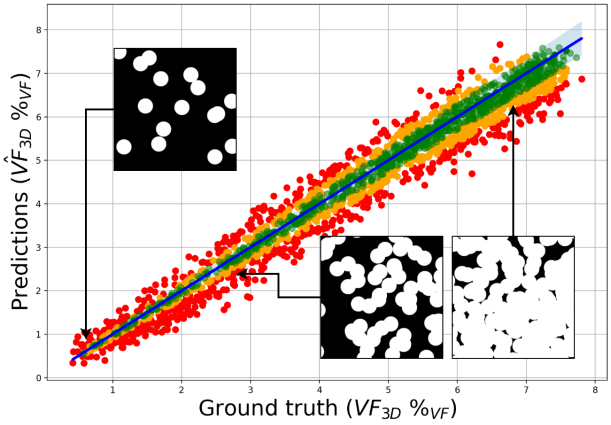

Fig. 3. A visual comparison between the predicted values of the volume fraction $\hat{VF}$ and the ground truth VF. The first bisector indicating the expected values (blue line), and a 95% confidence interval (light green interval) are shown in the figure. Some examples of the tested images on the network are shown, for $VF = 0.57\%_{VF}$, $2.74\%_{VF}$, and $7.12\%_{VF}$ respectively.

B. Estimating particle radius in a monodispersed field

Here we consider a mono-distribution of particles in a continuous phase (i.e. all particles have the same size in a single image). The objective is to estimate the radius of these particles in a dense area, where the size varies from one image to another. Therefore, the second network, CNN^{R}, is used. The training of the network is done using generated images, where the number of particles in each image is fixed almost at 300 particles, and the radius varies between 20 and 72 pixels from one image to another. Meaning that the volume fraction of the dispersed phase varies from 0.9 $\%_{VF}$ to 45 $\%_{VF}$. The accuracy of the model is tested over 2000 synthetic images of the same law. The results are presented in figure 4. Here, we can see clearly that this network preformed correctly and did in fact retrieve all the needed information with high accuracy, where all the predictions over the test set had an error rate less than 5%.

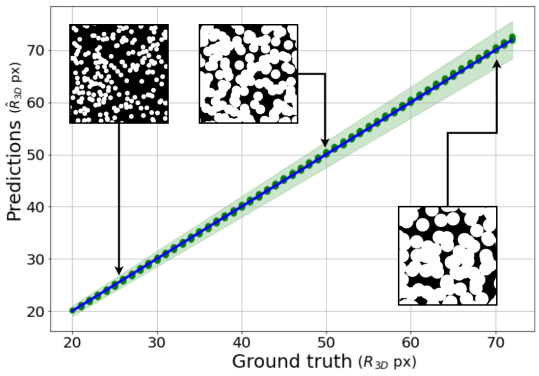

Fig. 4. A visual comparison between the predicted values of the radius $\hat{R}_{3D}$ and the ground truth R_{3D} (both units are in pixels). The first bisector indicating the wanted values (blue line), and a 95% confidence interval (light green interval) are shown in the figure. Some examples of the tested images on the network are shown, where $R_{3D} = 25,\ 50,$ and 70 respectively.

C. Estimating particles radius in a bidispersed field

The same approach is applied in this section, where the particles in the generated images are bi-distributed, meaning that they have one of two radii: a small radius that varies from 20 pixels to 46 pixels, and a large radius that varies from 47 pixels to 72 pixels. Both radii are equally distributed (i.e. 50% small radius, 50% large radius). Therefore, we employ a third network, CNN^{R2}, is trained to predict the two radii. The volume fraction of the dispersed phase in this case varies from $5.8\%_{VF}$ to $29.04\%_{VF}$. The accuracy of the model is tested over 2000 synthetic images of the same law. The results are presented in figure 5. Despite the number of points that lies outside the confidence interval for small $R1_{3D}$, the network did have a very good estimation of the wanted values, where 95% of the tested images had an error rate less than 5%.

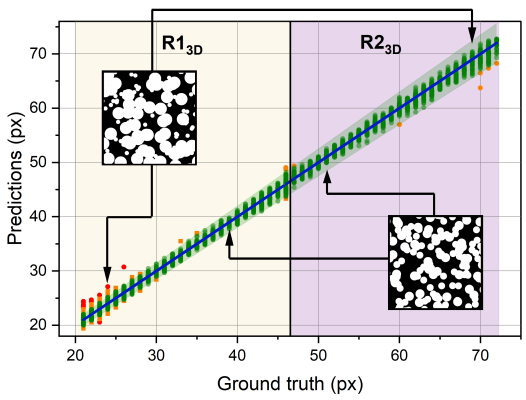

Fig. 5. A visual comparison between the predicted values of the radii $(\hat{R1}_{3D}, \hat{R2}_{3D})$ and the ground truth $(R1_{3D}, R2_{3D})$ (both units are in pixels). The first bisector is shown indicating the wanted values (blue line), in addition to a 95% confidence interval (light green area). The figure is split in two areas: the light yellow area indicating the predictions of the first radius $R1_{3D}$, and the light purple area indicating the predictions of the second radius $R2_{3D}$. Some examples of the tested images on the network are also shown, where $(R1_{3D}, R2_{3D}) = (24, 69)$, and $(40, 51)$ respectively from left to right.

D. Estimating the particle size distribution parameters

The final network will be used to estimate the mean radius μ. The radii are distributed following a truncated normal law with a mean μ that varies from one simulation to another, a standard deviation σ that is fixed, and lies within the interval $[20, 72]$. Three case studies where considered in this part of the study:

- CNN_1^{μ}: $25 \leq \mu \leq 70$ and $\sigma = 2\%$
- CNN_2^{μ}: $25 \leq \mu \leq 70$ and $\sigma = 10\%$
- CNN_3^{μ}: $25 \leq \mu \leq 70$ and $\sigma = 50\%$

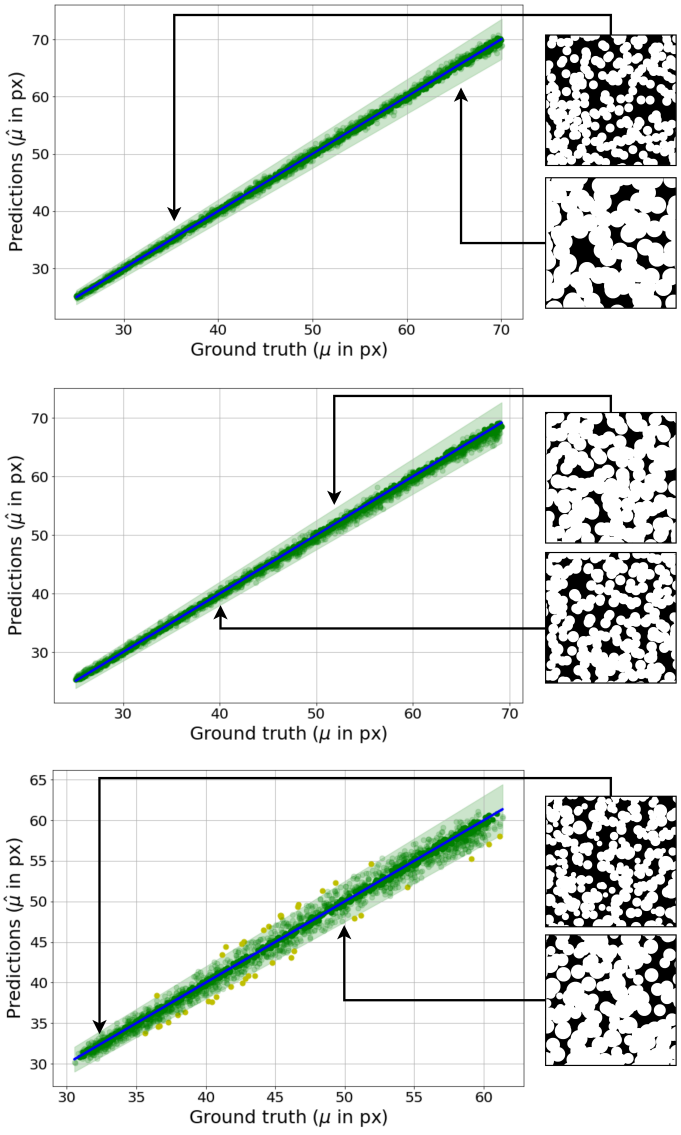

Fig. 6. A visual comparison between the predicted values of the mean radius, $\hat{\mu}$, and the ground truth μ (both units are in pixels). Each figure represents the prediction for each case study, where μ = 2%, 10%, and 50% from top to bottom respectively. The first bisector indicating the expected values (blue line), and a 95% confidence interval (light green interval) are shown in the three figures. Some examples of the tested images on the network are also shown, where μ = 36.12, 65.8, 52.56, 40.81, 32.9, and 50.14 respectively from top to bottom.

Each network is validated using 2000 synthetic images of the same law. The three networks preformed correctly, where all of the predicted values had an error rate less than 5% (expect for the third case where 16 cases lied outside of the confidence interval). A slight variance appears between the results of one network to another that can be remarked as the standard deviation of the radii increase from one case to another. Therefore, indicating that the results are less accurate in more complicated cases where the particle distribution is polydisperse (i.e. the radii change from one particle to another).

E. Application on real images

In an attempt to evaluate the efficiency of our approach on a real application, a mixture of calibrated PMMA particles and brine are considered. The salt concentration is tuned to adjust the liquid (i.e. H_2O) and the solid densities in order to prevent buoyancy effects. In this case we have access to precise properties of the dispersed phase. Three populations of mono-distributed spherical particles, with a radius $r = 1.59$ mm, are constructed:

- *Population 1:* the volume fraction is equal to $2.51\%_{VF}$ (i.e. nearly 1530 particles are present in 1 litre of the continuous phase),
- *Population 2:* the volume fraction is equal to $4.89\%_{VF}$ (i.e. nearly 3060 particles are present in 1 litre of the continuous phase),
- *Population 3:* the volume fraction is equal to $7.16\%_{VF}$ (i.e. nearly 4590 particles are present in 1 litre of the continuous phase).

1) Image acquisition and processing: Acquisitions were performed using a backlight setup including a Photron CMOS camera type FASTCAM Mini UX100, a bitelecentric lens and a collimated green light source to optically suppress perspective effects. This type of setup allows to project directly the apparent bead area on the sensor with a high contrast. The images have a size of 1280x1024 (cropped to 980×980 for processing purpose), where 100 pixels are equal to 2.60 mm. The acquisition rate is of 50 frames per second, with an exposure time of 1/81920 s allowing to prevent motion blur. A set of 4365 images was acquired for each experiment. The preprocessing of images consists of a simple binarization (Figure 7).

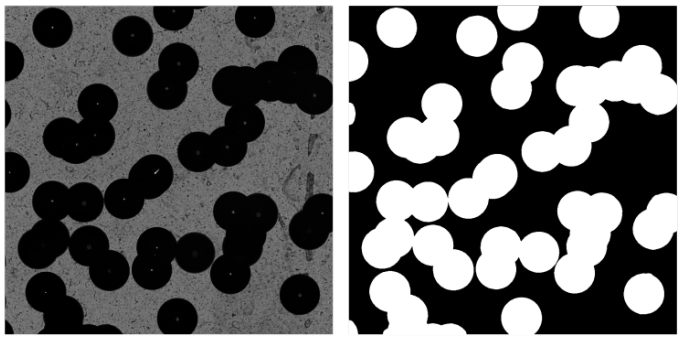

Fig. 7. The result of the binarization process (right) on an acquisition image (left) from the population 2.

2) Testing the network on the real images: Here, we test real images on our network. A CNN^{VF} that predicts the volume fraction is trained with synthetic images of the same type as the real images, i.e. same particle size and shape. The results are shown in figure 8 as a scatter of the estimated values. On the left, the ground truth is represented by a bold line and the variance of the values by an error bar. On the right, histograms of the predicted values are represented showing the distribution of the results. Therefore, we can clearly see that the model is capable of retrieving the correct volume fraction for the real images.

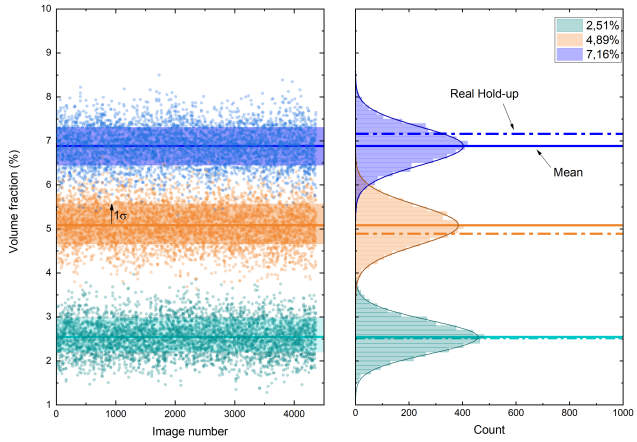

Fig. 8. Distributions of the predicted volume fraction on the acquisitions. The left figure indicates the predicted volume fraction for the 3 acquisitions of more than 4000 images. Each image is portrayed by a single point: green points represents for the first population with a volume fraction of $2.51\%_{VF}$, orange points highlights the second population with a volume fraction of $4.89\%_{VF}$, and the blue points highlights the third population with a volume fraction of $7.16\%_{VF}$. The mean of the predicted values is represented with thick lines, in addition to a confidence interval. On the right, three histogram of the predicted values are represented. The experimental volume fraction (i.e. hold up) is indicated by a dashed line.

F. Overall performance

In order to evaluate the overall performance of a regression model where the outcome is continuous, then evaluating the error rate is the right choice. Here, we use the mean squared error (MSE) as our evaluation metric. The MSE represents the average squared difference between the estimated values $\hat{VF}_{3D}$ and the actual value, VF_{3D}. In this case, the average loss on the observed data set. The unit of the volume fraction is the percentage $\%_{VF}$. The performance of all the tested networks are presented in the following tables.

Metrics	CNN^{VF}	CNN^{R}	CNN^{R2} (R1,R2)
MSE	$0.15\%_{VF}$	0.14 px	(0.5 px, 0.6 px)
Metrics	CNN_1^{μ}	CNN_2^{μ}	CNN_3^{μ}
MSE(μ)	0.25 px	0.48 px	0.95 px

TABLE I

THE MSE RATE OVER 2000 TESTED IMAGES FOR ALL THE FOUR NETWORKS.

Regarding the test over the real images, the following table shows the evaluation of the method.

Metrics ($\%_{VF}$)	Population 1	Population 2	Population 3
Real	2.51	4.89	7.16
Estimated (mean $\pm$ std)	2.54$\pm$0.38	5.08$\pm$0.45	6.88 $\pm$0.43
MSE	0.15	0.24	0.27

TABLE II

RESULTS OF THE ESTIMATION PROCESS ON THE MONODISPERSED CALIBRATED PMMA SPHERES POPULATION.

G. Conclusion and prospects

In this paper, a machine learning based approach has been proposed in order to retrieve 3D information of a field of particle from their 2D projections. The approach mainly consists on training a convolutional neural network with 2D projection of synthetic images labeled with 3D information. It proved to be rather efficient in terms of accuracy and time consumption over simple cases, where it was validated using both synthetic and real images which contain strictly spherical particles. These promising results mainly justify our interest in this type of algorithm, and encourage us to extend the research on more complex cases (check figure 9) where the particles may have more complicated shapes (e.g. ellipsoids, deformed particles, spherical caps, etc.).

To further our research, we intend to establish a more realistic 3D stochastic model by introducing more physical reality, such as the correlation between the size and shape of particles, the heterogeneity of a flow and the morphological deformation of each particle [Theodon et al., 2021]. Once the model is set up, the CNN network should also be modified to eliminate all types of overfitting (e.g. the CNN^{λ}), and be capable of understanding and estimating harder 3D properties from 2D projections.

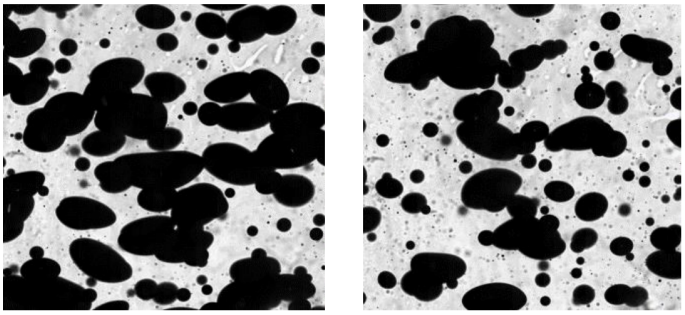

Fig. 9. An example of a real two phase flow.

REFERENCES

[Bian et al., 2013] Bian, Y., Dong, F., Zhang, W., Wang, H., Tan, C., and Zhang, Z. (2013). 3d reconstruction of single rising bubble in water using digital image processing and characteristic matrix. *Particuology*, 11(2):170–183.

[Farhan et al., 2013] Farhan, M., Yli-Harja, O., and Niemistö, A. (2013). A novel method for splitting clumps of convex objects incorporating image intensity and using rectangular window-based concavity point-pair search. *Pattern Recognition*, 46(3):741–751.

[Haas et al., 2020] Haas, T., Schubert, C., Eickhoff, M., and Pfeifer, H. (2020). BubCNN: Bubble detection using faster RCNN and shape regression network. *Chemical Engineering Science*, 216:115467.

[Hosokawa et al., 2009] Hosokawa, S., Tanaka, K., Tomiyama, A., Maeda, Y., Yamaguchi, S., and Ito, Y. (2009). Measurement of micro bubbles generated by a pressurized dissolution method. *Journal of Physics: Conference Series*, 147:012016.

[Khalil et al., 2010] Khalil, A., Puel, F., Chevalier, Y., Galvan, J.-M., Rivoire, A., and Klein, J.-P. (2010). Study of droplet size distribution during an emulsification process using in situ video probe coupled with an automatic image analysis. *Chemical Engineering Journal*, 165(3):946–957.

[Kim and Park, 2021] Kim, Y. and Park, H. (2021). Deep learning-based automated and universal bubble detection and mask extraction in complex two-phase flows. *Scientific Reports*, 11(1).

[Langlard et al., 2018a] Langlard, M. D., Al-Saddik, H., Charton, S., Debayle, J., and Lamadie, F. (2018a). An efficiency improved recognition algorithm for highly overlapping ellipses: Application to dense bubbly flows. *Pattern Recognition Letters*, 101:88–95.

[Langlard et al., 2018b] Langlard, M. D., Lamadie, F., Charton, S., and Debayle, J. (2018b). A 3D stochastic model for geometical characterization of particles in two-phase flow applications. *Image Analysis & Stereology*, 37(3):233.

[Lau et al., 2013] Lau, Y., Deen, N., and Kuipers, J. (2013). Development of an image measurement technique for size distribution in dense bubbly flows. *Chemical Engineering Science*, 94:20–29.

[Li et al., 2020] Li, J., Shao, S., and Hong, J. (2020). Machine learning shadowgraph for particle size and shape characterization. *Measurement Science and Technology*, 32(1):015406.

[Riquelme et al., 2013] Riquelme, A., Desbiens, A., Bouchard, J., and del Villar, R. (2013). Parameterization of bubble size distribution in flotation columns. *IFAC Proceedings Volumes*, 46(16):128–133.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science*, pages 234–241. Springer International Publishing.

[Seredkin et al., 2020] Seredkin, A., Plokhikh, I., Mullyadzhanov, R., Malakhov, I., Serdyukov, V., Surtaev, A., Chinak, A., Lobanov, P., and Tokarev, M. (2020). Pattern recognition for bubbly flows with vapor or gas-liquid interfaces using u-net architecture. In *2020 Science and Artificial Intelligence conference (S.A.I.ence)*. IEEE.

[Theodon et al., 2021] Theodon, L., Eremina, T., Dia, K., Lamadie, F., Pinoli, J.-C., and Debayle, J. (2021). Estimating the parameters of a stochastic geometrical model for multiphase flow images using local measures. *Image Analysis &amp Stereology*, 40(3):115–125.

[Zhang et al., 2012] Zhang, W.-H., Jiang, X., and Liu, Y.-M. (2012). A method for recognizing overlapping elliptical bubbles in bubble image. *Pattern Recognition Letters*, 33(12):1543–1548.