Kernelized Lagrangian Particle Tracking

Yin Yang, Dominique Heitz

Experiments in Fluids manuscript No.
(will be inserted by the editor)

Kernelized Lagrangian Particle Tracking

Yin Yang[1] · Dominique Heitz[1]

Received: date / Accepted: date

Abstract In this article, we present a novel Lagrangian particle tracking method derived from the perspective of the tracking-by-detection paradigm that has been adopted by many vision tracking tasks. Under this paradigm, the particle tracking problem consists of first learning a function (the tracker) that maps the target particle's image projection backwardly to its possible position inferred from its precedent tracking information. The target particle's actual position is then detected by simply applying the learned function to particle images captured by cameras. We also propose to solve the function learning problem using kernel methods. The proposed method is therefore named Kernelized Lagrangian particle tracking (KLPT). The current state-of-art LPT approach Shake-The-Box (STB), despite equipping a highly efficient image matching and shaking-based optimization procedure, tends to be trapped by local minimum when dealing with datasets featuring complex flows or larger time separations. KLPT can overcome these optimization difficulties associated with significant prediction errors since it features a highly robust function learning procedure combined with an efficient linear optimization technique. We assessed our proposed KLPT against various STB implementations both on synthetic and real experimental datasets. For the synthetic dataset depicting a turbulent cylinder wake-flow at Re3900, we focused on studying the effects of particle density, time separation, and image noises. KLPT outperformed STB in all cases by tracking more particles and producing more accurate particle fields. This performance gain, compared to STB, is more prominent for the dataset with larger seeding density, time separation, and more noise. For real experimental data on an impinging jet flow in a water tank, KLPT can capture longer tracks and provides more detailed flow reconstruction at highly turbulent regions than STB. Overall, comparison to STB shows significant improvements in accuracy (lower reconstruction positional error) and robustness (more and longer tracks). We finally show the results of KLPT on the 1st LPT challenge datasets. Our algorithm has achieved state-of-the-art performance with particle images up to 0.08 ppp (particles per pixel).

Keywords Lagrangian Particle Tracking · Particle Tracking Velocimetry · Kernel methods · Data assimilation

Y. Yang
E-mail: yin.yang@inrae.fr

D. Heitz
E-mail: dominique.heitz@inrae.fr
[1] INRAE, UR OPAALE, Rennes Cedex F-35044, France

1 Introduction

In fluid dynamics research and applications, reconstructing the 3D flow through imaging particles driven by the flow, followed by inferring spatiotemporal flow structures from particle images, is crucial to investigate the small scale flow phenomena related to turbulence.

The algorithms used to infer spatiotemporal flow fields from particle images can be classified under two categories. The first approach is the tomographic particle image velocity (TomoPIV) that centers on a tomographic reconstruction of particles on a 3D voxel basis (Elsinga et al. 2006). The Eulerian velocity field is then obtained by applying the classic cross-correlation technique, inherited from the standard 2D PIV technique to 3D intensity voxels. Thus the central task is to solve the intensity at (virtual) voxel coordinate from particle images. We can tackle this problem iteratively using algebraic reconstruction technique (ART), MART and SMART algorithms (Scarano 2013). The TomoPIV technique has gained considerable success since the last decade due to its robustness and its capability of dealing with particle images of high seeding density (particle per pixel, ppp around 0.05). Nevertheless, the TomoPIV technique still suffers from several flaws such as many ghost particles, relatively large particle positional error, and is computationally demanding.

The second approach is the 3D particle tracking velocimetry (3D-PTV). Also referred to as the Lagrangian particle tracking (LPT), it aims at following particle positions through time. The 3D-PTV has been proposed since the 1990s (Maas et al. 1993, Malik et al. 1993). However, this approach has not reached the same level of popularity and maturity as the PIV technique because 3D-PTV can only tackle images of relatively low seeding densities.

Inspired by the success of TomoPIV, Wieneke (2012) proposed an iterative particle reconstruction (IPR) approach that features an active image matching scheme. This matching procedure is carried out after the classical 3D-PTV operations (2D peak detection, stereo matching, and 3D triangularization) to refine further the particle's 3D position and facilitate the ghost detection. After one particle's rough location is known through triangulation, the IPR procedure optimizes the best estimation of particles' positions by iteratively matching their image space projections through an Optical Transfer Function (OTF) model with their observational records captured by cameras. IPR shares with TomoPIV the idea of 'matching' an imaging model with the image target. However, IPR reconstructs a field in particle representation rather than in voxel representation as for the TomoPIV. IPR can handle densely seeded flows up to 0.05 ppp while reaching a similar accuracy compared to TomoPIV.

IPR has paved the way for Shake-The-Box (STB) designed by Schanz et al. (2016), the current state-of-art method in LPT. This algorithm has shown its superiority in higher reconstruction accuracy, lower ghost particle occurence, and more economical computational resources demandings than TomoPIV. The problematics of ghost particles, resulting from the increased particle correspondence ambiguities associated with particle images of high seeding densities, is mostly resolved by the STB method. Consequently, STB can handle dense particle images (ppp reaching and beyond 0.1) and yield almost ghost-free particle fields. The power of STB lies in the coupling of IPR with a particle trajectory predictor. Hence, the starting position to perform IPR for further optimization is not from triangulation but from the predictor. Such modification leads to two significant implications: first, for flows where the particle position predictor is reasonably good, the initial error at the beginning of the IPR is considerably lower than that yielded by the triangularization. Second, for densely seeded flows (ppp > 0.1), STB tackles the problem in a progressive way leading to more accurate particle positions and fewer ghost particles. The triangulation on the full raw particle image set is determined to fail due to particle overlapping and correspondence ambiguities with high densities. Whereas for STB, as more and more particles are tracked, fewer and fewer particles need to be identified on the residual image using 3D-PTV.

Schanz et al. (2016) has shown that STB is well suited for time-resolved data depicting medium-low speed flow configurations. Besides, the image quality has huge impact on the positional accuracy obtained by STB. Recent developments concentrated on extending the power of STB to other types of data. For example, Novara et al. (2016, 2019) proposed the multi-pulse STB to deal with high-speed flow applications for which time-resolved data is not available. Tan et al. (2020) introduced the pruning algorithm to remove ghost particles and applied the STB method to blurred particle images. Other studies are more application-specific concerning a particular sub-module of the whole LPT workflow, such as the calibration procedure of Schröder et al. (2020) and Huhn et al. (2018). However, very few works exist to improve the core tracking ability of STB.

Still, STB can fail under two circumstances. First, even STB cannot tackle very high ppp levels. Schanz et al. (2016) has reported that the algorithm fails for image sets of ppp above 0.125. During our participation in the first LPT and DA challenge, we have also found STB implemented in Davis 10, which is the only available commercial STB implementation on the market, can only track a small portion of particles for data sets of very high ppp (0.16 and 0.2). The final challenge results revealed that STB algorithms of LaVision and DLR have succeeded in tracking dataset of very high ppp (0.16 and 0.2). However, it is uncertain whether or not the added components for STB by LaVision and DLR compared to the commercial version, which helps to handle high ppp dataset, can be easily generalized to other datasets. Second, in the cases of **sparse temporal data** and/or **data extracted from complex flows**, the Wiener filter or the polynomial-based predictor systematically leads to larger particle positions prediction errors. Consequently, we have found that STB is less effective and can fail because any local minimum can trap the simple "Shaking" optimization algorithm. From an optimization point of view, the STB scheme requires the cost function to be relatively locally smooth to perform well. This high level of smoothness can not be guaranteed when the predictor failed to provide a good starting point for data with large-time separation or local complicated flow structure. The consequences of tracking failure are either one track is terminated prematurely, or one particle is identified on the wrong track. Schanz et al. (2016) proposed two ways to alleviate those divergence issues. One way is by adding an *initial shake* stage prior to the *fine shake* stage, so as to provide a *warm start* for the *fine shake* stage. The other way is by adding post-processing routines. For example, the outlier removal procedure allows the mistracked particle to be removed if the particle velocity clearly deviates from its neighboring velocity. Those adhoc techniques are crucial since they can largely increase the quality of the reconstructed field. However, since they are adhoc models, they can be challenging to apply to other datasets depicting different flow scenarios. For example, even the simplest adhoc model requires some hyper-parameters to be fine-tuned against data, while the fins-tuning process can be troublesome and expensive in real experiments. Besides, some adhoc models can not be generalized to other flows as they target a specific type of dataset. We argue that the extensive use of adhoc models reflects the failures of the core tracking ability of the LPT tracker. By employing more powerful tracking schemes, an LPT tracker should rely less on the adhoc techniques and handle more experimental cases without too much interference from experimental practitioners. To obtain such a robust LPT scheme, we advocate a thorough revise of the particle tracking optimization problem.

This paper provides a new interpretation of the particle tracking problems from the perspectives of both machine learning (ML), and data assimilation (DA) approaches. LPT is essentially a Computer Vision (CV) task. Since most of the best methods for solving CV tasks already benefit from ML methods, we think that a formalism of LPT using ML helps to introduce more advanced ML techniques into our community. The DA technique infers the best state by combining the observed state with a dynamic model. Compared to a pure ML approach, DA puts more emphasis on controlling the dynamic model. Since particle image sequences contain real particle motions whose transport is underlaid by the flow governing equations, it is essential to consider equally both information to estimate physically consistent flow fields. This new interpretation proposed in this paper leads to solving the LPT task based on the empirical risk minimization paradigm. The resulting algorithm is named the Kernelized LPT (KLPT). The word 'kernelized' stands for the famous Kernel trick that can be employed to solve unknown functionals that minimize the empirical risk. Following the presentation of the framework, we concentrate on improving the accuracy and the robustness of the particle state estimation procedure, therefore boosting the performance of the algorithm for **sparse temporal data** and/or **data extracted from complex flows**. We also establish the link between our framework and an ensemble DA method (Yang et al. 2015, 2018).

Section 2 recapitulates the STB working principles from a statistical point of view. This 'new' formulation allows us to understand better why STB outperforms other 3D-PTV/LPT schemes. More importantly, we also identify the primary source of error associated with STB and propose corresponding strategies to overcome this limitation. In the following section, we show that by employing the KLPT scheme, the performance is mostly boosted in accuracy and robustness compared to STB. Due to the increased robustness of our scheme, we do not need to consider some ad-hoc tricks to deal with challenging tracking scenarios. We show the results of applying KLPT both on a synthetic dataset and a real experimental dataset against other STB schemes in section 4 and 5, respectively. We also show in section 6 the results of applying KLPT on the 1st LPT challenge datasets.

2 STB's anatomy

2.1 State space formulation

We start formulating the LPT problem using the (latent) state-space formulation dealing with the time-resolved data. First, we recall that the central task of LPT boils down to find the same particle at different acquisition times k over image frames: $0 \to K-1$ where K is the number of total frames. However, we do not observe the particle positions directly. Instead, we observe them through images. Thus we need to model the data generation process that links the *latent variable*, 3D particle position $\mathbf{X}$, to the gray-scale values in images I captured by cameras. We can write the observation equation at time k as:

$$I_p^{k,i} = \mathcal{I}_p^i(\mathbf{X}_p^k, E_p^k) + \boldsymbol{\epsilon}_p^k, \tag{1}$$

where $\mathcal{I}_p^i$ is the observation model for particle p, i is the camera index, and $\boldsymbol{\epsilon}$ is the uncertainties associated with this observation model. In analogy to the active-projection model used in Wieneke (2012) and Schanz et al. (2016), $\mathcal{I}_p$ is a combination of the camera model M, and the Optical Transfer Function (OTF), also known as Point Spread Function (PSF). M links the 3D position X_p in object space to 2D position x_p in image space. The OTF takes the particle intensity E_p in addition to X_p yielding the pixel intensity values in the vicinity of 2D position x_p. Here we denote I_p^i an image patch extracted from a small square box located around the particle's projection x_p on the recorded image I_{rec}^i. If the camera model and the OTF are well-calibrated, we expect that the image samples create by $\mathcal{I}$ resembles the one captured by the camera. So the objective of LPT is to find $\mathbf{X}_p^k$ that minimized the following loss function measuring the similarities between the image records I and the image samples $\mathcal{I}$:

$$J(\mathbf{X}_p^k) = \frac{1}{2}\sum_i ||I_p^{k,i} - \mathcal{I}_p^i(\mathbf{X}_p^k, E_p^k)||^2. \quad (2)$$

A slightly different version is used in Wieneke (2012) as an iterative reconstruction method following the stereoscopic reconstruction procedure.

$$J(\mathbf{X}_p^k) = \frac{1}{2}\sum_i ||I_{res+p}^{k,i} - \mathcal{I}_p^i(\mathbf{X}_p^k, E_p^k)||^2. \quad (3)$$

where $I_{res+p} = I_{res} + \mathcal{I}_p$ and $I_{res} = I_{rec} - \sum_p \mathcal{I}_p$. This version is more robust against overlapping particles. Note that the original notion of image matching can be traced back to the TomoPIV algorithm, where it solves the 3D voxel intensities.

The particles are subject to the dynamics of the fluids. More specifically speaking, if we know the 3D position X of a particle p at time $k-1$, then we can model its evolution to time k by a discrete Markovian nonlinear state mapping operator φ:

$$\mathbf{X}_p^k = \varphi_k(\mathbf{X}_p^{k-1}) + \boldsymbol{\xi}_p^k, \quad (4)$$

where $\boldsymbol{\xi}^k \sim \mathbb{N}(0, \mathbf{Q}^k)$ denotes the uncertainties associated with the dynamical model φ. This dynamic model φ is natively recursive, which means we must know *a priori* the initial condition $\mathbf{X}_p^0$. In practice, the phase of obtaining the distribution of the entire particle set $\mathbf{X}^0$ is called particle and track initialization that is generally done before particle tracking process.

2.2 Why STB works?

Above latent state-space formulation is nothing but an abstraction of the core algorithm that is implied in Schanz et al. (2016) in an unseen and heuristic manner. Indeed, we can express the core algorithm of STB under the same formulation by noting that:

- The state predictor operator φ in Schanz et al. (2016) takes the form of the Wiener filter or the polynomial-based filter. Those filters, in which the future state depends on more than the closet history state, can be understood as a higher-order Markov chain model: $\mathbf{X}_p^k = \sum_{h=1}^n \lambda^h \mathbf{X}_p^{k-h}$;
- The optimization approach, used in IPR/STB for finding $\mathbf{X}_p^k$ minimizing the cost function (2), belongs to a group of model-based derivative-free trust region optimization methods. To formalize STB scheme, we employ a quadratic form $m_k(\delta X_p)$ that locally approximates the original cost function $J(\delta X_p)$ in terms the particle's X coordinate while keeping Y and Z coordinates invariant:

$$m_k(\delta X_p) = c + g^T\delta X_p + \frac{1}{2}\delta X_p^T G \delta X_p. \quad (5)$$

  Then with 3 sample points, we can uniquely determine the coefficients c, g, G, which are all scalar with respect to a single particle. We need to compute I_{part}^i 7 times for each particle. Compare to the time dedicated to computing the image projection, the computing time used to determine the extreme of the above quadratic function is trivial;
- The intensity E_p is rescaled in a subsequent procedure:

$$\hat{E}_p^k = E_p^k h[\frac{\sum(I_{res+p})}{\sum(\mathcal{I}_p(\hat{\mathbf{X}}_p^k, E_p^k))}]. \quad (6)$$

Before the STB approach, most multi-view 3D-PTV methods have been built on two pillars: particle reconstruction (2D peak finding, stereoscopic reconstruction) and particle tracking. Depending on the context, the particle tracking procedure can be done either before or after the stereoscopic reconstruction procedure. The stereoscopic reconstruction task is primarily resolved using the epipolar geometry constrain with fully-calibrated cameras. Due to the lack of discriminative intra-particles features, this reconstruction process can create many false-positive results, commonly known as ghost particles, that do not exist in reality. The tracking procedure, either done in image space or object space, can be cast as an optimization problem that searches for the best particle correspondences that minimize some *kinematic* measurement metrics (Ouellette et al. 2005).

STB uses the state predictor to approximatively reconstruct the particle in 3D object space for the current frame. Then the reconstructed particle field is further refined through optimizing some *image-based* metrics (e.g. intensity as in Eq. (2)). Such a scheme has several advantages compared to the classic multi-views-based 3D-PTV approach. Firstly, the complete stereoscopic reconstruction for each frame is avoided. In STB, the whole process is radically accelerated as the Wiener/polynomial-based filter and the optimization of (2) through (5) are substantially cheaper. The stereoscopic reconstruction is still needed, but only to reconstruct the new particles and those untracked ones. Secondly, the found position is supposed to be more accurate than those with stereoscopic reconstruction solely. Each step in the stereoscopic reconstruction algorithm, e.g., peak finding, stereo-matching, and triangulation, can introduce errors. In STB, if the cameras are well-calibrated, the remaining primary source of error can be attributed to the OTF calibration. When the cameras and the OTF are all well calibrated, the effectiveness of the STB method, therefore, largely hinges on the accuracy of the state predictor. With a good predictor, the STB method starts with an initial particle field reconstructed using either 3D-PTV or Tomo-PTV and gradually converges by tracking more and more particles. The algorithm is considered converged if all but new particles into the measurement domain are successfully tracked. Without an accurate predictor, the mapped particle field remains far from the actual particle distribution such that the simple derivative-free optimization scheme (5) fails. Even worse, the wrongly predicted particle may overlap with a nearby particle resulting in a progressively diverging algorithm.

For sparse temporal data and/or data extracted from complex flows, the Wiener/Polynomial-based filter leads to large prediction errors. To resolve the issue, we can

1. replace the Wiener/Polynomial-based filter with a physical-based kinematic/dynamic predictor that reflects well the characteristics of the flow under investigation:
2. employ a more robust optimization scheme that is less sensitive to starting values so that we can still find the global minimum of (2) even with an erroneous predicted position.

Having reviewed both the advantages and challenges of STB, we detail and discuss our proposed method in the next section.

3 Kernelized LPT

In this section, we propose the framework of our method: the Kernelized LPT. We begin with a succinct presentation of the kernel methods, and then we discuss why we choose the kernel methods over others.

3.1 Tracking-by-Detection with Kernel

Again we recall that the central task of LPT is to track the same particle over image frames. From a visual tracking perspective, LPT aims to follow the small local patch I_p^i on each camera corresponding to the same particle p. Due to the nature of the particle image, we can assign the particle's 3D coordinate $\mathbf{X}_p$ to the local patch set $\mathbb{I}_p$ denoting the set containing I_p^i for all cameras:

$\mathrm{f} : \mathbb{I}_p \to \mathbf{X}_p,$

where f is the mapping function from $\mathbb{I}_p$ to $\mathbf{X}_p$. Unlike classic 3D-PTV where f can be explicitly modeled through 2D peak finding and stereoscopic reconstruction procedures, we intend to learn f from data sets composed of N samples $(\mathbb{I}_p^j, \mathbf{X}_p^j)_{j=1}^N$, where each sample pair $(\mathbb{I}_p^j, \mathbf{X}_p^j)$ represents a possible state of particle p. So the problem of tracking comes down to match a position with image patches $\mathbb{I}_{p*}^{rec}$ on a new record frame. Note that the subscript p^* reflects that the image record patch $\mathbb{I}_{p*}^{rec}$ is extracted from the neighboring pixels of a prior position $\mathbf{X}_p^*$ because the actual position is unknown. After f is learned, we can compute the corresponding particle coordinate as: $\hat{\mathbf{X}}_p = \mathrm{f}(\mathbb{I}_{p*}^{rec})$.

For the current LPT study, we obtain the random samples in the following way. First we add Gaussian noise σ to the position at frame $k-1$: $\mathbf{X}_p^{j,k-1} = \mathbf{X}_p^{k-1} + \sigma_p^{j,k-1}$, then we propagate each sample from time $k-1$ to k using formula (4). We may need to pertube the position traced back to $k-h$ where h is the order of the fitted filter if necessary. Intensity needs to be also perturbed to obtain distinctive samples. Finally, we rely on formula (1) to generate image patches $\mathbb{I}_p$.

Interpreting sample for 2D-PTV The notion of *sample* is inherited from the machine learning community, which can be more clearly clarified under 2D-PTV/PIV context. For example, suppose we know the local patch I_p in the form of a rectangle of size $m \times n$ at the previous frame, and we want to track this target at the current frame. Then we can collect samples in the neighborhood of I_p. The sampling process can be done either randomly or densely. The random way is to get a patch the same size as I_p centered at a different random position. The dense way is to translate I_p in x or y direction with a displacement of 1 pixel at a time. Then we can assign a label y to each sample. The value of y is arbitrary (e.g. binary or continuous) and can be tailored to different applications.

3.1.1 Tracking principle

We propose to learn f using the empirical risk minimization paradigm. If given a set of particles' 3D positions $\mathbf{X}_p$ and their corresponding projections on several cameras $\mathbb{I}_p$, we can train a tracker $\mathrm{f}(\mathbb{I}_p)$ that minimizes the following regularized empirical risk:

$$R(\mathbf{f}) = \sum_{j=1}^{N} Q(\mathbf{X}_p^j, \mathbb{I}_p^j, \mathbf{f}(\mathbb{I}_p^j)) + \lambda ||\mathbf{f}||_{\mathcal{H}_K}^2, \quad (7)$$

where Q is some measurement function. A popular choice is the $L2$ norm $Q(\mathbf{X}_p, \mathrm{f}(\mathbb{I}_p)) = ||\mathbf{X}_p - \mathrm{f}(\mathbb{I}_p)||^2$.

Generally f is a nonlinear function(al) that links I_p to $\mathbf{X}_p$. Given the nature of our problem, $\mathrm{f} \in \mathbb{R}^d$ is necessary a vector-valued function. In the following, we show how to solve for f by minimizing R using a kernel. The procedure is slightly different from the similar procedure dealing with a scalar-valued function found on any machine learning textbook. A more detailed discussion on kernels for vector-valued function can be found in Micchelli & Pontil (2005), Álvarez et al. (2012). The introduction of the regularization term in (7) reduces possible solution sets and increases the optimization stabilities. The regularizer is expressed in terms of the dot product of functions in the Reproducing Kernel Hilbert Space (RKHS). Interestingly, given this kind of regularizer, according to the Representer theorem, the solution takes a rather special form:

$$\mathbf{f}(\mathbb{I}) = \sum_{j}^{N} \bar{\bar{\kappa}}(\mathbb{I}_p^j, \mathbb{I})\boldsymbol{\alpha}^j, \tag{8}$$

where $\bar{\bar{\kappa}} \in \mathbb{R}^{d\times d}$ is a matrix-valued kernel function that compares the similarities between a sample $\mathbb{I}_p^j$ and another candidate. $\boldsymbol{\alpha}^j \in \mathbb{R}^d$ is a weighting vector.

Kernel The role of a kernel is to measure the *similarities* between two samples. We list a few forms of kernel functions in the appendix. For current study, we define the matrix-valued kernel function as

$$\bar{\bar{\kappa}}(\mathbb{I}^i, \mathbb{I}^j) = \kappa(\mathbb{I}^i, \mathbb{I}^j)\mathbf{C}, \tag{9}$$

where $\kappa(\mathbb{I}^i, \mathbb{I}^j)$ is a scalar-valued kernel function and $\mathbf{C}$ is a $d \times d$ matrix encoding the output field information. The simplest case is to consider $\kappa(\mathbb{I}^i, \mathbb{I}^j) = (\mathbb{I}^j)^T\mathbb{I}^i$ as the dot-product kernel and $\mathbf{C}$ as an identity matrix. In this case, the output field is uncorrelated. The solution form (8) corresponds to the dual solution of problem (7). Alternatively the primal solution can be expressed as $\mathbf{f}(\mathbb{I}) = \mathbf{w}^T\boldsymbol{\phi}(\mathbb{I})$ where $\boldsymbol{\phi}$ maps $\mathbb{I}$ to some (nonlinear) feature space. Such a form of the primal solution changes the original nonlinear functional minimization to a linear one in w. The kernel function can be defined as the dot product of two transformed samples in feature space: $\kappa(\mathbb{I}^i, \mathbb{I}^j) = \langle\boldsymbol{\phi}(\mathbb{I}^i), \boldsymbol{\phi}(\mathbb{I}^j)\rangle$. The dual solution formulation has the advantage of avoiding any explicit feature mapping process because the solution is just a function of the kernel function.

Learning Inserting the above form about f into (7) leads to an optimization problem in a large flat vector $\boldsymbol{\alpha} = [(\boldsymbol{\alpha}^1)^T, \cdots, (\boldsymbol{\alpha}^N)^T] \in \mathbb{R}^{dN}$. Here we directly give the solution of $\boldsymbol{\alpha}$ through minimizing $R(\boldsymbol{\alpha})$:

$$\boldsymbol{\alpha} = (\bar{\bar{K}} + \lambda\mathbf{I}_{dN})^{-1}[(\mathbf{X}_p^1)^T, \ldots, (\mathbf{X}_p^N)^T]^T, \tag{10}$$

where $\bar{\bar{K}}$ is the Gram matrix with the kernel function $\bar{\bar{\kappa}}_{ij}(\mathbb{I}^i, \mathbb{I}^j)$ as its block element.

Detection So now, given the camera images $\mathbb{I}_{p*}^{rec}$, we can directly compute $\hat{\mathbf{X}}_p$:

$$\begin{aligned}\hat{\mathbf{X}}_p =& \mathbf{f}(\mathbb{I}_{p*}^{rec}) \\ =& \big(\bar{\bar{\kappa}}(\mathbb{I}_p^1, \mathbb{I}_{p*}^{rec}), \cdots, \bar{\bar{\kappa}}(\mathbb{I}_p^N, \mathbb{I}_{p*}^{rec})\big) \\ & (\bar{\bar{K}} + \lambda\mathbf{I}_{dN})^{-1}[(\mathbf{X}_p^1)^T, \ldots, (\mathbf{X}_p^N)^T]^T.\end{aligned} \tag{11}$$

Incremental approach The solution (11) depends on the inversion of the Gram matrix $\bar{\bar{K}}$. The stability of the solution can be compromised in cases where the original gray level $\mathbb{I}$ is used as the feature $\phi(\mathbb{I})$, because the elements of $\bar{\bar{K}}$ risk of having close values, leading to an ill-conditioned matrix. A common and simple technique to improve the stability of solution (11) is to apply the so-called data centering to the original feature space (Shawe-Taylor & Cristianini 2004). In this paper, we adopt a similar strategy by first centering the particle's samples around its mean: $\mathbf{X}_p^{'j} = \mathbf{X}_p^j - \bar{\mathbf{X}}_p$, where the mean $\bar{\mathbf{X}}_p = 1/N\sum_{j=1}^N \mathbf{X}_p^j$. Then the corresponding centered image projection is represented by $\mathbb{I}_p^{'j} = \mathbb{I}_p^j - \bar{\mathbb{I}}_p$ where $\bar{\mathbb{I}}_p$ is the projection of $\mathbf{f}(\bar{\mathbf{X}}_p)$. The tracking solution takes the same form as (11) while replacing the original data sample pairs by the centered one $(\mathbb{I}_p^{'j}, \mathbf{X}_p^{'j})$ and the camera images $\mathbb{I}_{p*}^{rec}$ by its residual $\mathbb{I}_{p*}^{res} = \mathbb{I}_{p*}^{rec} - \bar{\mathbb{I}}_p$. The resulting optimal increment $\hat{\mathbf{X}}_p^{'}$ needs to be added to the mean $\bar{\mathbf{X}}_p$ to retrieve finally the particle's optimal position finally. Because the centered feature vector $\mathbb{I}_p^{'}$ measures the relative intensity variation instead of the absolute intensity levels, the resulting kernel function values are therefore more distinct to each other, leading to a more stable solution.

Our method is thus preceded as kernelized to pay attribute to the revolutionary work of Kernelized Correlation Filter (KCF) (Henriques et al. 2014) from which our initial idea is inspired. However, KCF is a 2D general visual tracking scheme, while our approach is a 3D object-space tracker that leverages the kernel function execution in image space and features a matrix-valued kernel function as well.

3.2 Why kernel method?

We choose the kernel method over other methods mainly due to its flexibility, as explained in the following.

3.2.1 Joint intensity optimization

It is possible to estimate the intensity of one particle E_p in addition to its 3D location. By constructing the augmented data sample pairs:

$$(\mathbb{I}_p^j, ([(\mathbf{X}_p)^T, E_p]^T)^j)_{j=1}^N,$$

we can rely on the same forms of the formula (7), (8), (10) and (11) by replacing $\mathbf{X}_p$ with $[(\mathbf{X}_p)^T, E_p]^T$ and noting that the dimension of unknown vector is now $d = 4$.

3.2.2 Flexibility with kernel choice

The famous kernel trick allows us to employ any proper kernel function to compute $\hat{\mathbf{X}}_p$. According to our experience, a simple dot product kernel $\langle \bullet, \bullet \rangle$ provides satisfactory results for raw image intensity features $\mathbb{I}_p$. Still, it is straightforward to switch to other relevant kernel functions more adapted to the dataset. For example, The simple dot product kernel between two local raw image patches can be cast as a cross-correlation operation. In the same spirit, we can consider using the normalized cross-correlation (NCC), which can potentially perform better if the intensity fluctuates a lot between two subsequent steps. When considering different kernel functions, we must make sure it is a valid kernel. Interested readers can find in Shawe-Taylor & Cristianini (2004) or Schölkopf & Smola (2002) a strict mathematic formulation on how to prove if a function is a valid kernel. Here we give some guidance on how to justify a valid kernel through some simple observations. First, the kernel function has to be symmetric: $\kappa(x, y) = \kappa(y, x)$. Second, since the kernel function is equivalent to an inner product in some feature space, it should satisfy $\kappa(x, y) > 0$. Finally, the resulting Gram matrix is positive definite. These kernels are positive definite kernels, and they measure the similarities between two input samples. A dissimilarity measure, e.g. the squared sum-differences (SSD) $||x - y||^2$, can be generally transformed to a similarity measure by $g = e^{-\alpha||x-y||^2}$ with an arbitrary scalar coefficient α (Haasdonk & Bahlmann 2004). In this way, distance-related function can also be employed as valid kernel function for (10) and (11).

3.2.3 Working with local image features

In multi-view PTV, we deal mostly with raw image intensities. However, it is straightforward to replace those raw intensity features with more meaningful local image features. Changing of kernel function as done in section 3.2.2 implies a feature space transformation without explicitly transforming. It remains the only way to map the original feature to an infinite feature space since only the dot-product is needed. However, some datasets can be beneficial to use an explicitly designed feature extractor Φ. For example, the census transformation (Zabih & Woodfill 1994) can transform a single-pixel intensity to a vector of bits encoding the relative intensity variations of the target pixel w.r.t its neighbors. This process is widely used to encode the intensity's local variations into the feature space. After the feature transformation, we may also need to adjust the kernel function. For example, the Hamming distance can be employed to compute the (di)similarities of two transformed census samples. Since it is a distance function, we need to alter the final function form such that it becomes a valid kernel function suited for our algorithm. Another interesting alternative is to use features that are learned by a deep neural network (DNN). To that end, we need to train a deep feature descriptor based on data encapsulating the local image patch with a positive and a negative pair, respectively similar to Balntas et al. (2016). Introducing those kinds of features can also shed light on the new sample generation process.

3.2.4 Obtain optimized track simultaneously

The link between the kernel and the differential equations has been established in Steinke & Schölkopf (2008). In equation (7), the regularizer is expressed in terms of the dot product in the RKHS. An alternative way is to express the regularization term as the Euclidean dot product $||\mathbf{R}\mathrm{f}||^2$, where $\mathbf{R}$ is some regularization operators. It has been shown in Steinke & Schölkopf (2008) that the two regularization forms are identical under some conditions. We can indeed infer the kernel function through the regularization operator. In other words, employing this kind of regularization operator $\mathbf{R}$ pre-selects the feature space to which f belongs. However, this feature space is also an RKHS equipped with kernel κ. This relationship has profound implications. For one thing, the fluid dynamic often takes the form of a differential equation that *regulates* both spatially and temporally the evolution of the flow (including the tracers). For individual particle, considering only the temporal regularity, the function f now relates $\mathbb{I}_p^{k,k+l-1}$ to $\mathbf{X}_p^{k,k+l-1}$. The upper script $k, k + l - 1$ indicates the variable encapsulates its values at different time snapshots. We know from equation (4) that the 3D positions $\mathbf{X}_p$ at different times are subject to the temporal constraint imposed by φ. It is always possible to find the corresponding $\mathbf{R}$ that regulates the output of f: now being one track instead of one position. In appendix A, we provide detailed mathematical formulations on obtaining $\mathbf{R}$ from dynamical constraints. Then we can compute the equivalent kernel function from $\mathbf{R}$. Finally, it is possible to obtain simultaneously the optimal track of particle p ranging from frame k to $k+l-1$. This strategy yields the *smoothed* particle trajectory directly. We use the term 'smoothed' in the strict sense indicating any particle positions $\mathbf{X}_p$ on the obtained trajectory are optimally conditioned on all observed particle images from frame k to $k + l - 1$. A track filtering component is also discussed in the STB paper (Schanz et al. 2016) that features a third-order smoothing B-spline. Under this terminology, smoothing, aiming at decreasing the roughness of the temporal trajectory, is used to de-noise the noisy raw tracks. Thus our strategy is different from the track filtering component. Our track reconstruction strategy is similar to the time-segment assimilation technique (González et al. 2019) that tried to recover the temporal evolution of the Eulerian velocity field governed by the vorticity transport equation and Poisson equation from particle tracking data measured at multiple time instants. This added feature to KLPT would be indeed helpful and deserves its own dedicated paper to fully shine. We limit ourselves to this general discussion to show that KLPT is extremely powerful and readily extendible to cover more advanced topics regarding either the dynamics or the image data.

In this article, we focus on the algorithm based on the core scheme discussed in section 3.1 enhanced by the joint intensity estimation strategy discussed in section 3.2.1. We do not extend our algorithm to cover other more advanced features discussed in section 3.2 as the simple raw feature plus the dot product (cross correlation) kernel has already yielded excellent results for our test datasets that can hardly be outperformed by employing more sophisticate features or kernel functions. Besides, KLPT with the simple raw feature and the dot product kernel requires the lowest computational time, only slightly larger than STB.

3.3 Improvements compared to Ensemble-based Data Assimilation approach (EnDA) (Yang et al. 2018)

Our strategy in this paper is different from an earlier version of KLPT discussed in Yang et al. (2018). In Yang et al. (2018), the EnDA solution is formulated by using the ensemble-based method to solve equation (2) with an additional regularization term $||\mathbf{X}_p^k||^2$. The context of this strategy is still sample-based so we stick to the same notations defined in section 3.1.

The basic idea of EnDA is first to linearize the original non-linear cost function around the mean prior state $\bar{\mathbf{X}}_p$, then we search for an increment $\mathbf{X}'_p = \mathbf{X}_p - \bar{\mathbf{X}}_p$ instead of the original $\mathbf{X}_p$. Such manipulation leads to a solution similar to the Kalman filter with the state's covariance matrix being approximated by its ensemble covariance counterpart defined over the centered samples $[\mathbf{X}_p^{\prime 1}, \cdots, \mathbf{X}_p^{\prime N}]$. Interested readers can find additional details of the EnDA approach in Yang et al. (2015, 2018). Here we directly give the EnDA solution as:

$$\hat{\mathbf{X}}'_p = [\mathbf{X}_p^{\prime 1}, \cdots, \mathbf{X}_p^{\prime N}][(\vec{\mathbb{I}'})^T \vec{\mathbb{I}'} + \lambda \mathbf{I}_N]^{-1} (\vec{\mathbb{I}'})^T \mathbb{I}_{p*}^{res}, \tag{12}$$

where $\vec{\mathbb{I}'}$ is a vector collecting the centered data $[\mathbb{I}_p^{\prime 1}, \cdots, \mathbb{I}_p^{\prime j}, \cdots, \mathbb{I}_p^{\prime N}]$.

Interestingly, we can show in the appendix that the solution given in (12) is indeed a particular case of formula (11). The interrelations between the kernel solution following the empirical risk minimization paradigm and the EnDA solution following the data assimilation/hidden Markov model is probably more profound and beyond the scope of this paper. However, in the following, we highlight a few differences and links between the two approaches. Firstly, we are free to choose any relevant kernel function in (11) while in (12) only the simplest dot-product can be considered. Secondly, the DA approach emphasizes on the perspective of controlling the parameters of the dynamical model φ. In contrast, the kernel approach interprets the dynamical model as a regularization term since it emphasizes more on the data. Lastly, for the stereoscopic system of 3D flow visualization, the data sample $\mathbb{I}_p^j$, which is used to either approximate the error covariance for EnDA approach or formulate the data pairs for kernel approach, is not readily available in reality but generated through first the dynamical model (4), then the observation model (1). These models' validities hinge on whether the dynamical model φ explicates well the nature of the flow and whether the camera model and the OTF model are well-calibrated. For our applications, many good dynamical models do exist that can be tailored for different flow configurations. The calibration of the stereoscopic camera system has been studied extensively over the years and can reach a calibration error lower than 0.1 px. The OTF calibration is a relatively new topic, and generally, it is not very difficult to obtain a good accuracy even for blurred particle images Schanz et al. (2012). Using synthetic-based models to infer pertinent knowledge from *real* data, although seemingly paradoxical and inconsistent, is adopted by many ML approaches. Indeed many state-of-art deep learning methods are trained on synthetic data but can generalize well to real data.

3.4 Summary of KLPT Algorithm

STB includes several other building blocks to ensure that the whole LPT scheme works on stereoscopic image sets. We also implemented those building blocks in addition to the core tracking scheme of KLPT. Below we list some of those important modules and briefly mention our implementing details:

- The calibration of the camera model and the OTF model is done according to Tsai (1987), Soloff et al. (1999) and Schanz et al. (2012), respectively;
- The particle initialization procedure builds the initial tracks for the first few frames (usually 4). There are various ways to obtain this initial field. We have implemented two methods. When the Eulerian velocity field for the first few frames is roughly known, we use a nearest-neighboring-based tracklet creation algorithm. Otherwise, we use a four-frame-based tracker (Ouellette et al. 2005);
- The procedure for adding particles features an IPR-like procedure to retrieve the positions of

new particles entering the domain as well as the positions of lost particles. Then a nearest-neighboring-based tracklet creation algorithm is used to produce tracklet of short length (≤ 4); A particle is removed when its position exits the domain or when its rescaled intensity drops under a certain threshold.

- The post-processing techniques featuring e.g. the track filtering algorithm and the outlier removal procedure (Schanz et al. 2016). We do not implement any track filtering algorithm. The outlier removal algorithm is implemented for real experiments but not used for synthetic tests.

The whole KLPT algorithm in summarized in the following Algorithm 1.

Algorithm 1: KLPT

```
1: Calibrate cameras, record particle image sequences
   $I_{\text{rec}}$ ranging from 1 to $K$, calibrate OTF.
2: Choose adequate kernel functions $\bar{\bar{\kappa}}$.
3: Build initial track $\mathbf{X}_{1\sim 4}$ from frame 1 to 4 using
   triangulation or IPR.
4: At frame $k-1$, for each particle, generating samples
   $\mathbf{X}^{j}_{k-1,p}$ by adding perturbations to $\mathbf{X}_{k-1}$.
5: Predict each sample using equation (4), note that the
   intensity follows a constant evolution: $E_k = E_{k-1}$.
6: Generate sample pairs $(\mathbb{I}^{j}_{p}, \mathbf{X}^{j}_{p})$ using (1) and crop
   $\mathbb{I}^{rec}_{p*}$ for camera records.
7: Obtain $\hat{\mathbf{X}}_{k,p}$ through equation (11).
8: Proceed IPR on residual image to identify
   overlapping/new particles and build tracklet.
9: Check convergence criteria, if met, set $k := k+1$ and
   go to step 4, otherwise resample dataset $(\mathbb{I}^{j}_{p}, \mathbf{X}^{j}_{p})$ and
   go to step 7.
```

4 Synthetic data assessment

4.1 Synthetic particle images creation and experiment configuration

To assess our proposed method quantitatively against the STB scheme, we have created synthetic images based on datasets of the Eulerian velocity of a Large Eddy Simulation (LES) (Parnaudeau et al. 2008). The data depicts a turbulent wake-flow past a circular cylinder at Reynolds number equal to 3900. We chose one snapshot of the velocity field in a subdomain of size $6D \times 1.9D \times 1D$ of the full simulation domain. The subdomain center is located $6D$ downstream from the cylinder axis in the wake flow (cf. Figure 1).

The initial distribution of locations and intensities of the virtual particles are randomly generated within the subdomain. Each particle is then transported according to its velocity calculated from trilinear interpolation of the velocity vectors on the nearest neighboring LES grids. We use the first-order Euler method to solve for the predicted particle position. Although this case is stationary, the flow field is characterized by fast spatial fluctuations induced by turbulence, which is drastically different from the smoothly and slowly varying velocity field used in other studies. When the time separation between two consecutive snapshots ΔT_{obs} is relatively short compared to the LES simulating time step δt, the polynomial-based predictor remains effective. However, the prediction errors become higher with larger ΔT_{obs} since the temporal distribution of observations cannot capture the fast spatially changing velocity. Under such circumstances, we are inclined to have larger positional errors and lose track of more particles. To evaluate our method performance and STB for different temporal resolutions, we employed two different time separation levels for ΔT_{obs} ($3\delta t$ and $15\delta t$ respectively). The mean particle displacement was around 2 pixel in the case of $\Delta T_{obs} = 3\delta t$ while the case $\Delta T_{obs} = 15\delta t$ led to a larger displacement of nearly 10 pixels.

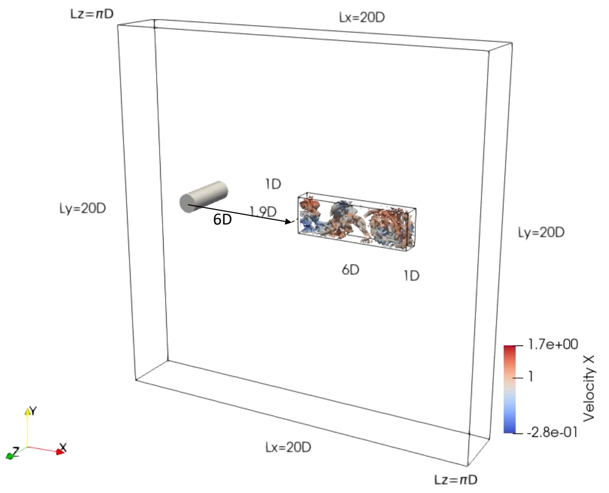


Fig. 1: Instantaneous vorticity isosurface colored by stream velocity in a subdomain of the LES of Parnaudeau et al. (2008).

The virtual particles transported by the flow are projected onto four virtual cameras (1280×800 pixels each) under a cross-like pose configuration with a rotation of $\pm 30°$ around x and y axis, respectively. All cameras are pre-calibrated using both the polynomial mapping model and pinhole model. In terms of the OTF parameter (Schanz et al. 2012), we employed a uniform two-dimensional Gaussian form resulting in a constant particle diameter around 2 px. The tested seeding density varied from 0.05 to 0.1 ppp.

4.2 Evaluation strategy

In section 3.4, we have discussed some critical, non-tracking modules that play vital roles in determining the flow reconstruction quality. Naturally, we need to separate their impacts on the reconstruction results if we want to emphasize the impact of the core tracking scheme. Thus in this work, we propose to compare our KLPT scheme to both an academic version of STB code implemented in our group (denoted by STB-in-house) and a commercial version of STB implemented in the software Davis 10 (STB Davis).

For Davis 10, the camera model and the OTF model are re-calibrated from synthetic data sets. The camera model obtained in Davis resulted in a calibration error of 0.01 px, which was at the same level as our calibrated camera model for KLPT/STB-in-house. Unlike the camera model, the errors associated with the calibrated OTF parameters of Davis are not directly available. We also noticed that the OTF calibration module in KLPT/STB-in-house is highly accurate because the structure of the module is implemented the same as the form of the proper OTF function. On the one hand, we slightly modified the calibrated OTF parameters for KLPT/STB-in-house to provide a fair comparison. On the other hand, we adopted a trial-and-error strategy such that the modulated OTF parameters (intensity and radius) for Davis yielded the lowest possible reconstruction errors.

To eliminate the impact of different particle initialization realizations on the results, we proceed as follows: first, we run the STB program in Davis after importing our synthetic image and the camera mapping function. After the STB run in Davis, we extract the particle field of frames 1 to 4 from Davis's output field. Then we use this four-frame particle field to initialize STB-in-house and KLPT.

Still, some differences remain between every two algorithms:

- Between the KLPT method and the STB-in-house: only the tracking scheme is different. The rest of the LPT approach, including the particle initialization and the particle adding/deletion, is common to both STB and KLPT. We do not employ any track filtering as well as outlier removal procedure for the synthetic data.
- Between the KLPT method and the STB-Davis: each method is entirely different such that the differences must be interpreted as the consequence of the whole algorithm, not just related to the tracking scheme.
- Between the STB-in-house and the STB-Davis: only the tracking scheme is the same (the STB-Davis is also based on Schanz et al. (2016) but is not open source).

Similar to Schanz et al. (2016), we compare the reconstructed particle fields from different methods against three metrics: the mean error of detected particles, the fraction of undetected particles, and the fraction of tracked ghost particles. Unlike Schanz et al. (2016), both our implementation and Davis do not keep track of the number of ghost particle unless it is on a track (length ≥ 4). Also, we observed that the levels of ghost particles for all three schemes were equally low for most of our tests. We considered that a reconstructed particle was detected if the reconstructed particle was found by searching the surroundings of every true particle within a certain radius (1 px) from the reference volume. Then we can compute the mean error of detected particles. Once an true particle was paired, it was removed from the reference pool. Then we count the size of the left reference pool as the number of undetected particles. We also considered that a tracked particle was a ghost if no true particle can be found by searching the surroundings of the tracked particle within a certain radius (1 px). We used 50 snapshots in total, and we checked the time evolution of the above quantities starting from the 5th snapshot.

4.3 Results on perfect image

We conducted the first experiment on noise-free images. We plotted the temporal evolution of mean positional error of detected particles, the fraction of undetected particles and the fraction of tracked ghost particles obtained by the three schemes at different ppp level for two time separation, $\Delta T_{obs} = 3\delta t$ (fig. 2) and $\Delta T_{obs} = 15\delta t$ (fig. 3) respectively. We also summarized in table 1 the values of the three metrics averaged over frame 40-44 for different approaches. The period ranging from 40-44 frames corresponds to the converged phase for most of our tests. We observe immediately that KLPT systematically outperforms the other two STB realizations for all tests.

The data sets generated with a small-time separation ($3\delta t$) is considered a more straightforward case for all algorithms since the predictor performs reasonably well. Under small time separation, the convergence processes of all three schemes are successful and take a similar amount of time-steps (immediately for 0.05 ppp and 5 time-steps 0.075 ppp). At 0.1 ppp, KLPT requires fewer time-steps to reach the convergence than STB-Davis (12 time-steps versus 17 time-steps). The convergence of STB-in-house at this high-density image is not evident as the mean error, the levels of undetected particles, and tracked ghosts still reduce even after the 45th frame. When time separation becomes larger ($15\delta t$), we observed different convergence behaviors starting at 0.075 ppp. At this particle density level, STB-Davis converge almost instantly after the initialization. KLPT and STB-in-house converge at the 10th time-steps, despite that STB-in-house is inclined to diverge after several snapshots being processed. The final values of fraction of undetected particles reached up to 17% which are one time larger than the one obtained by KLPT and STB-Davis. For higher particle image density (0.1 ppp), STB-Davis converges

Time separation		$3\delta t$			$15\delta t$		
Mean drift (px)		2			10		
PPP		0.05	0.075	0.1	0.05	0.075	0.1
Mean error of detected particles [px]	STB Davis	0.02992	0.05185	0.06890	0.03689	0.08791	0.1207
	STB-in-house	0.01862	0.02206	0.02915	0.02773	0.03819	0.2769
	KLPT	0.01729	0.01983	0.02181	0.02517	0.03841	0.0500
Fraction of undetected particles [%]	STB Davis	0.349%	0.382%	0.481%	7.083%	9.117%	16.36%
	STB-in-house	0.437%	0.881%	1.759%	4.275%	17.17%	81.75%
	KLPT	0.273%	0.331%	0.516%	1.931%	8.585%	16.98%
Fraction of tracked ghost particles [%]	STB Davis	0.01%	0.18%	0.163%	0.117%	0.423%	1.39%
	STB-in-house	0.01%	0.0533%	0.1805%	1.058%	3.119%	34.55%
	KLPT	0.01%	0.0147%	0.0245%	0.333%	1.011%	2.891%

Table 1: Comparison of KLPT and STB for perfect image sets averaged over snapshot 40-44

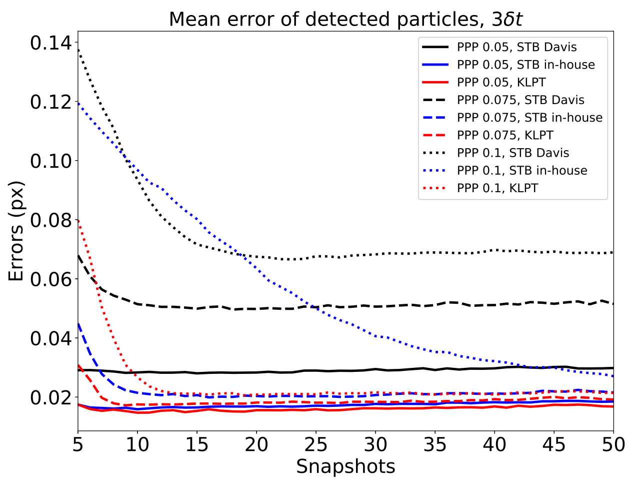
faster than KLPT (10 time-steps versus 15 time-steps). STB-in-house diverges entirely and can not yield anything useful.

The complicated behaviors of the three systems in terms of convergence have to be attributed to the particle adding procedure. Indeed, STB-Davis employs effective triangulation and tracklet creation procedures to introduce more particles into the system early after the initialization. The nature of triangulation and tracklet creation procedures do not depend on the central prediction-correction scheme, therefore, exempted from the effects introduced by large time separation. The failure of STB-in-house at high ppp shows that the same triangulation and tracklet creation functions used by both KLPT and STB-in-house are relatively inefficient. Still, KLPT can overcome this weakness and achieve the same level of convergence as STB-Davis.

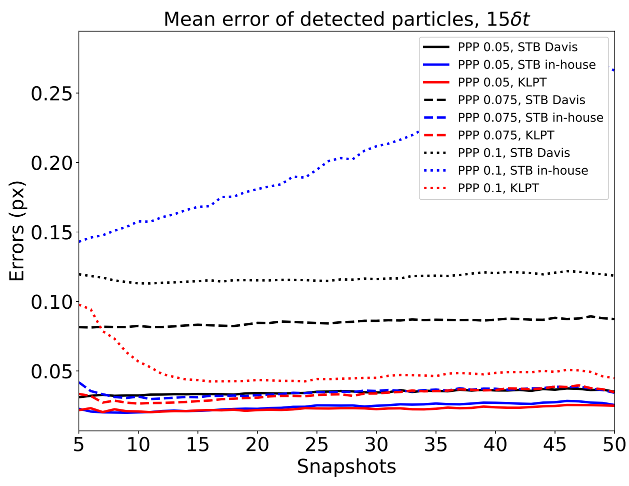
We also observe, that regardless of the time separation and the particle image densities, the levels of undetected particles and tracked ghosts, obtained from STB-Davis, are almost always similar to those obtained from KLPT (except for a fraction of undetected particles at 0.05 ppp). Nevertheless, the mean positional errors obtained from STB-Davis are substantially larger than KLPT, which leads to concrete and significant improvements of KLPT over STB-Davis. Unlike STB-Davis, STB-in-house showed some difficulties in convergence at medium-high ppp where STB-in-house yields an error level similar to KLPT but with the levels of undetected particles and tracked ghost much higher than KLPT and STB-Davis.

The distinct performance of STB-Davis and STB-in-house is more deeply connected to the trade-off between *robustness* and *accuracy*. Here we rely on more intuitive interpretations of the two concepts without defining them rigorously. The accuracy of the algorithm can be directly linked to the mean error of detected particles. A more robust algorithm tends to track more true particles and less false (ghost) particles as for robustness. Thus, the fractions of undetected particles and the tracked ghost are a good indicator of the algorithm's robustness. We give examples of how the natural trade-off between robustness and accuracy manifests itself during the particle tracking process. Particle overlapping is one of the reasons why a tracked particle becomes lost. Suppose one particle located on the overlapped region should belong to our target track, but the current algorithm failed to identify the particle for the target track. Then we modify the algorithm allowing the target track to continue on the overlapped particle. Inevitably this more robust version tends to track the wrong particle in the same overlapping region that in reality belongs to another track. Although the error of the mistracked particle stays the same as if its original track tracked it, the original track can terminate prematurely. Besides, the target track is likely to end at the next frame as the mistracked particle may lead to an unrealistic prediction. In all, this misinformation results in a degeneration of average errors.

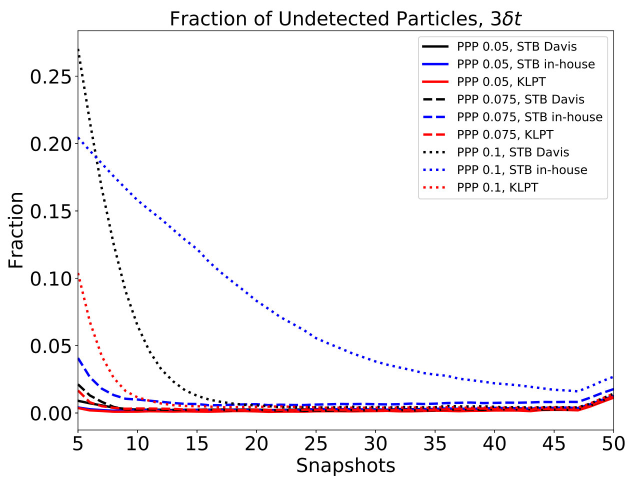
Since both STB implementations feature the same core tracking scheme, we find that the added robustness of STB-Davis comes from its other non-tracking modules (e.g. particle adding as mentioned above, outlier removal... see section 3.4). However, the added robustness is at the cost of sacrificing its accuracy compared to STB-in-house featuring a naive implementation of those non-tracking modules. Interestingly, KLPT, with the same naive strategy on non-tracking modules, yields the best results both in terms of accuracy and robustness. More precisely, the result obtained by KLPT is simultaneously on a par with or better than the best possible result obtained by each STB realization that maximizes either accuracy or robustness, respectively. Hence we can draw the following conclusions. Even though the non-tracking modules significantly influence reconstruction qualities, it is the core tracking scheme that determines the upper limit of the reconstruction quality. Bounded by this limit, one has the trade-off between robustness and accuracy. Besides, compared to STB, KLPT can increase this upper limit and yields virtually *better* particle fields. Finally, we did not need to introduce a very elaborate non-tracking scheme into KLPT to achieve the similar robustness as a commercial version of STB.



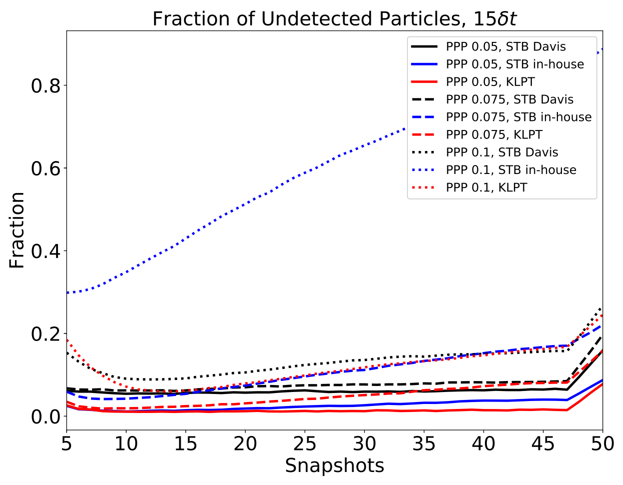
(a)



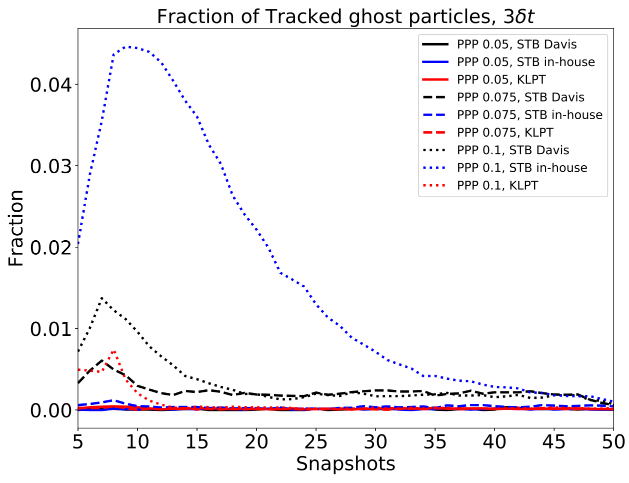
(a)



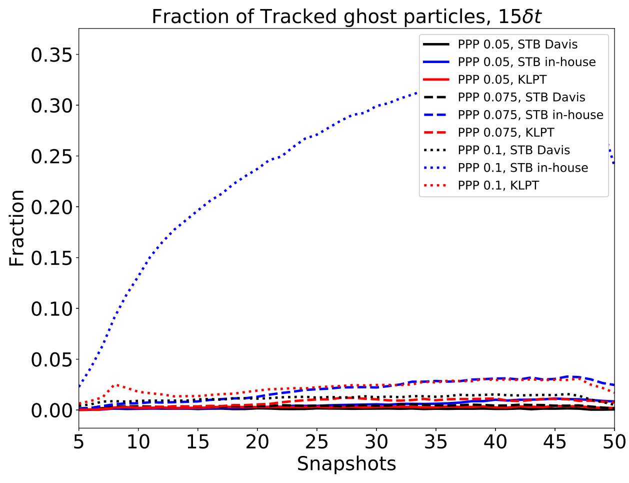
(b)



(b)



(c)



(c)

Fig. 2: Comparison of results produced by KLPT, STB-in-house and STB-Davis at time separation $3\delta t$ for different ppp levels: (a) mean positional error of detected particles; (b) fraction of undetected particles; (c) fraction of tracked ghost particles.

Fig. 3: Comparison of results produced by KLPT, STB-in-house and STB-Davis at time separation $15\delta t$ for different ppp levels: (a) mean positional error of detected particles; (b) fraction of undetected particles; (c) fraction of tracked ghost particles.

4.4 Effects of image noise

STB is sensitive to image noises because the optimization scheme evaluating the summed-square-differences (3) cannot distinguish between the image residual and the image noise. KLPT can be potentially more robust against image noises if a proper kernel function that uncorrelates the noises in (11) is employed. However, since we only test LPT with a simple dot product kernel function in this article, it is unclear whether KLPT can perform better than STB with noisy images. To assess the performance of two approaches, we conducted different series of tests by adding noises to synthetic image sequences at two particle densities ($ppp = 0.05$ and $ppp = 0.075$) with time separation $\Delta T_{obs} = 7.5\delta t$.

Besides a negative effect on the tracking scheme, image noises also affect the triangulation process for adding particles. For STB-Davis, it is impossible to separate the two effects.Thus, it is more consistent to compare the performance of KLPT and STB-in-house since the two algorithms feature the same triangulation implementations. We chose a medium time separation $7.5\delta t$ because testing on smaller time separation is relatively too simple for both approaches. In comparison, larger time separation leads to an early divergence for STB even without noise.

We generated three extra data sets by adding levels of noise to the original noise-free image data. The noise follows Gaussian distribution with zero mean and different variance. We characterized the noises using the peak signal-to-noise ratio (PSNR), which is defined as

$$\text{PSNR} = 10\log_{10}(\frac{d^2}{\text{MSE}}),$$

where d is the dynamics of the gray level of the signal, and MSE is the mean square error computed from the noisy image and the original noise-free image. The three noise levels correspond to 20 dB, 30 dB, and 40dB in PSNR. Note that the quality of the image is poorer with low PSNR values. Usually, we consider the images with PSNR of 30 to 40 dB are of good quality. Preprocessing is required for images with a PSNR of 20 dB to remove the noises. Nevertheless, similar to Schanz et al. (2016), we do not apply any image preprocessing techniques in advance to check the robustness of the tracking schemes against noises. We show in figure 4 the patches extracted from the same locations of the full image at four different noise levels.

We still evaluated the three metrics (mean positional error, levels of undetected particles and tracked ghost) to assess the performance of KLPT and STB. Figures 5 and 6 show the temporal evolution of the three metrics for ppp=0.05 and ppp=0.075, respectively. We also highlight in table 2 the values of three metrics averaged over frames 40-44 for different approaches.

We observe that KLPT (red curves) yields systematically better results than STB (blue curves). The two methods provide the most comparable results at ppp=0.05 and lower noise levels (noise-free and 40 dB). With a medium noise level at 30 dB, mean errors obtained by KLPT are 30% lower than those obtained by STB. The level of undetected particles of KLPT is 0.7%, which is 80% lower than STB and the level of tracked ghost of KLPT is 0.07%, which is 90% lower than STB. At the high noise level (20 dB), STB fails to converge with 83% of the true particles becoming undetected. It also produces 39% tracked ghost level, which suggests that nearly 1 out of 3 particles found by STB were ghosts. On the contrary, KLPT still produces acceptable results characterized by only 14% for undetected particles and 5% for tracked ghost particles. At higher particle image densities (ppp=0.075), figure 6 and table 2 exhibit a more prominent differences between KLPT and STB. KLPT can produce consistently good results with only a few particles undetected or tracked as ghosts. STB only provides acceptable results for medium noise level with 13% of true particles undetected and a mean positional error 70% higher than KLPT for low and medium noise levels. A high noise level (20 dB) marks the divergence of KLPT that captured only 50% of true particles. The tracked ghosts of KLPT (12%) are slightly larger than STB (9%). However, STB is only able to track 10% true particles. Such a low level indicates that most tracklets (length $\leq$ 3) reconstructed by STB tend to terminate prematurely as STB's optimization is corrupted by noises. Although the same procedure for adding tracklets was also implemented in KLPT, KLPT successfully developed a large portion of tracklets into valid tracks owning to the robustness of its optimization strategy.

4.5 Computational resource

In both STB and KLPT approaches, the computational time is dominated by the repeated execution of OTF. As a result, the CPU time per snapshot of KLPT with eight runs of OTF is about 15% more than that of STB, requiring seven runs of OTF. However, the scalability of KLPT is higher since an ensemble can be parallelized efficiently. On the memory side, KLPT is memory demanding, and the memory required by KLPT is a factor of sample members higher than STB under the same ppp level. Nevertheless, such memory usage is still affordable considering that any modern computer is equipped with several tens of GBs.

PPP		0.05				0.075			
PSNR		noise free	40 dB	30 dB	20 dB	noise free	40 dB	30 dB	20 dB
Mean error of detected particles (px)	STB	0.00685	0.02071	0.05582	0.3029	0.01872	0.03157	0.07549	0.373
	KLPT	0.00505	0.01589	0.03822	0.1413	0.01199	0.02245	0.04469	0.2051
Fraction of undetected particles (%)	STB	1.018%	1.259%	3.443%	83.726%	6.269%	6.897%	13.969%	90.168%
	KLPT	0.521%	0.52%	0.774%	14.767%	1.406%	1.455%	2.265%	50.634%
Fraction of tracked ghost particles (%)	STB	0.09%	0.197%	0.734%	39.083%	1.184%	1.416%	2.327%	9.477%
	KLPT	0.026%	0.054%	0.072%	5.304%	0.193%	0.194%	0.426%	12.41%

Table 2: Comparison of KLPT and STB for noisy image sets averaged over snapshot 40-44.

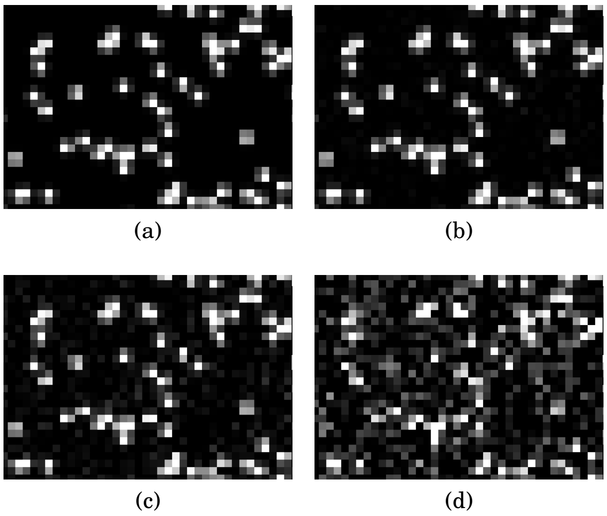


Fig. 4: Local patches of image samples of four noise levels (PSNR values): (a) noise-free; (b) 40 dB; (c) 30 dB; (d) 20 dB.

5 Experimental results

5.1 Experimental setup

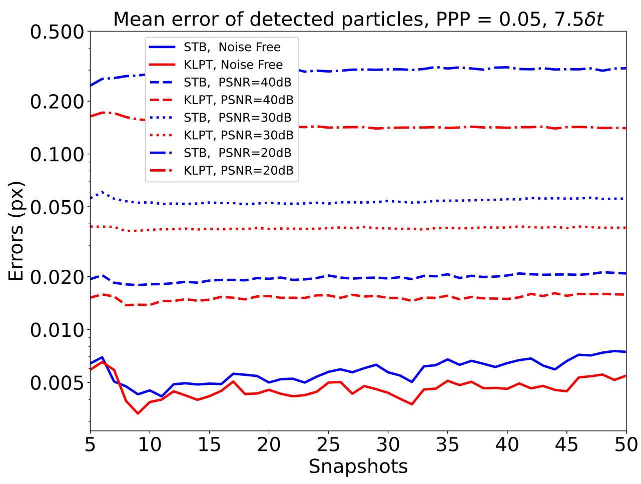
The experiments were carried out at the OPAALE research unit's fluid mechanics facilities, of the french national institut for agriculture, food and environment (INRAE) located at Rennes in France. The water tank and apparatus to generate the impinging jet flow was provided by the LaSIE from the University of La Rochelle in France. The details of the different jet flow configurations available with this facility can be found in Sodjavi et al. (2016). Originally, the measurements were designed and carried out to provide TomoPIV results for different type of nozzles, synchronized with wall electrodiffusion measurements. In the present study we use the particle image database to perform LPT for the jet flow with an axisymmetric convergent nozzle with an exit diameter $D = 7.8$ mm. The nozzle exit flow velocity was adjusted so that the Reynolds number based on D was equal to 1250. The distance between the nozzle exit and the ground was set to 24 mm ($\simeq 3D$).

Four cameras, Phantom Miro M310 ($1,280 \times 800$ pixels at max 3260 Hz) CMOS arrays (12-bit depth, 20 μm pixel size), were equipped with Nikon 105 mm Macro F2.8 focal length lenses set to F22, and arranged in the same plane at angles as shown in Fig. 7, in order to measure the wall impinged region of the flow. Prisms were used at the front glass window of the water tank to reduce the distortion created by the air-glass-water interfaces for each camera. The Scheimpflug condition was set with a double-angle system designed by Dantec to optimize the focus throughout the measurement volume.

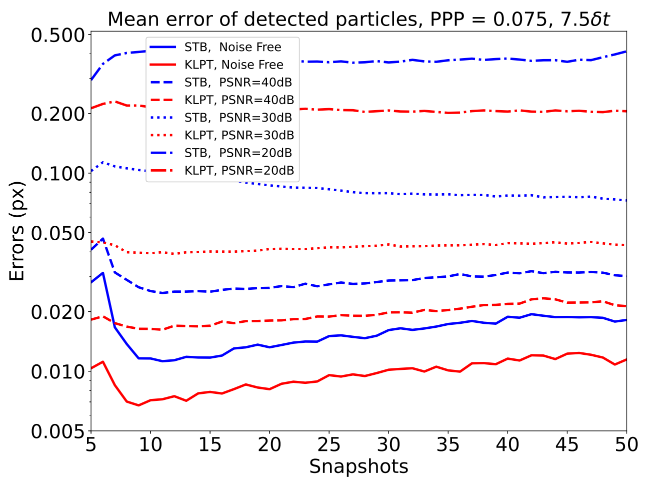
The flow was seeded, with hollow glass spheres of $9 - 13$ μm in diameter and 1.1 g/cm^3 in density, to a particle image density of approximately 0.03 ppp. Illumination was provided by a Litron LDY 300 laser with pulsed energy of 15mJ at 1kHz and 19mJ at 0.2kHz. The laser beam diameter of 4 mm was shaped into a sharply defined illuminated volume of section 35 mm $\times$ 24 mm, introduced from the top of the tank, thanks to a LaVision system composed of spherical and cylindrical lenses, followed by a knife-edge slit. Furthermore, the laser beam was back-reflected through the measurement volume by a mirror placed at the bottom of the tank.

The images used in the present study were acquired at a frequency of 500 kHz. The time delay between two laser pulses was set at 1000 μs. Two thousand image pairs were recorded with the LaVision Davis system. Calibration was performed with a two-depth level calibration target (Type 7 from LaVision), using the front and backside of the plate, translated through the illuminated volume. The volume self-calibration approach of Wieneke (2008) was performed to reduce the calibration error to less than 0.2 pixels after some iterations. The effective measurement volume was equal to 24 mm$\times 51$ mm$\times 35$ mm corresponding approximately to $3D \times 6.5D \times 4.5D$. Figure 8 shows a TomoPIV reconstruction of an instantaneous velocity field, using Davis 8 with the SMART algorithm followed by cross-correlation analysis with a final interrogation volume size of $32 \times 32 \times 32$ voxels at 50% overlap. In the following to assess our proposed KLPT approach, we cropped the images such that the resulting domain of interest was reduced to a volume of $16 \times 41 \times 20$ mm^3 corresponding approximately to $2.1 \times 5.3 \times 2.6D^3$, into the complex flow region near the wall.

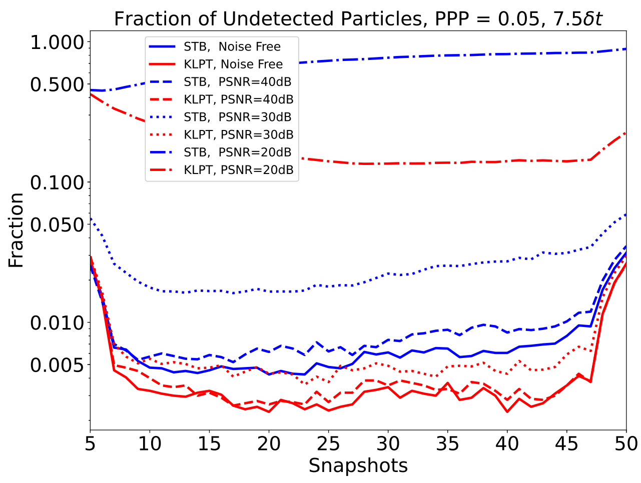
Similar to synthetic tests, we initialize KLPT run with the first four-frame tracks obtain by STB-Davis to provide a fair comparison. The camera model as well as the OTF are re-calibrated in KLPT



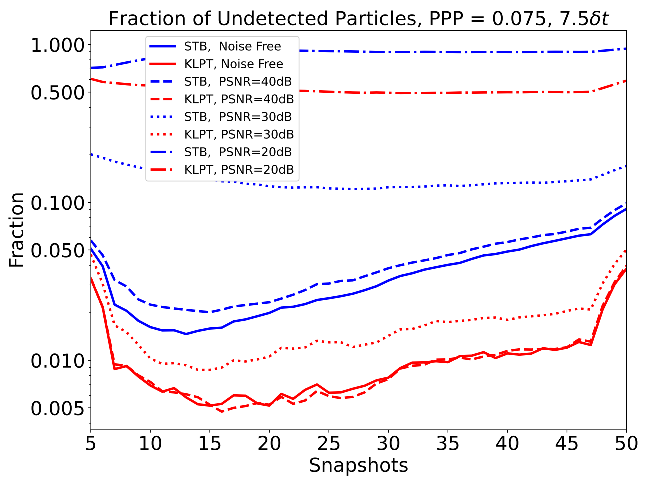
(a)



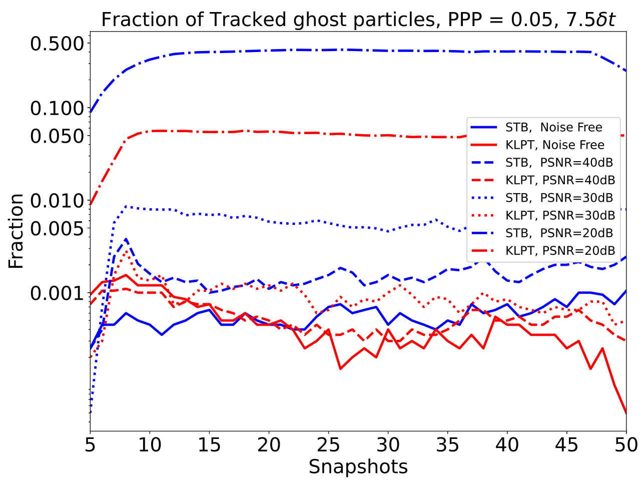
(a)



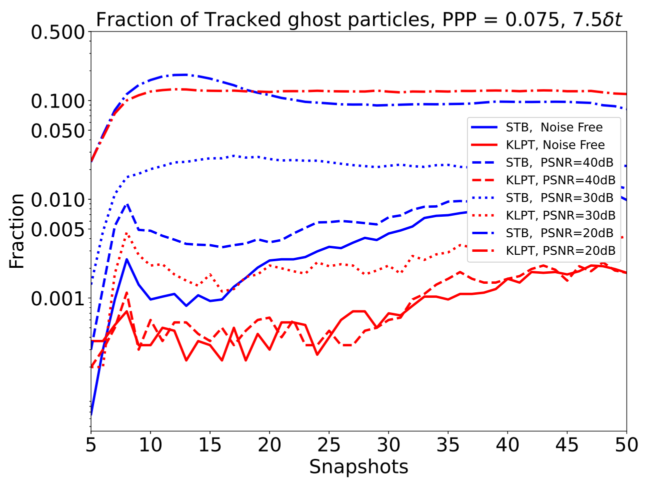
(b)



(b)



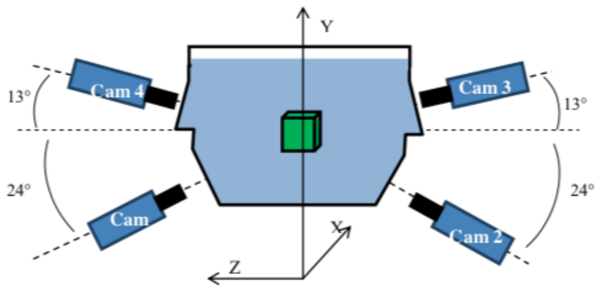
(c)



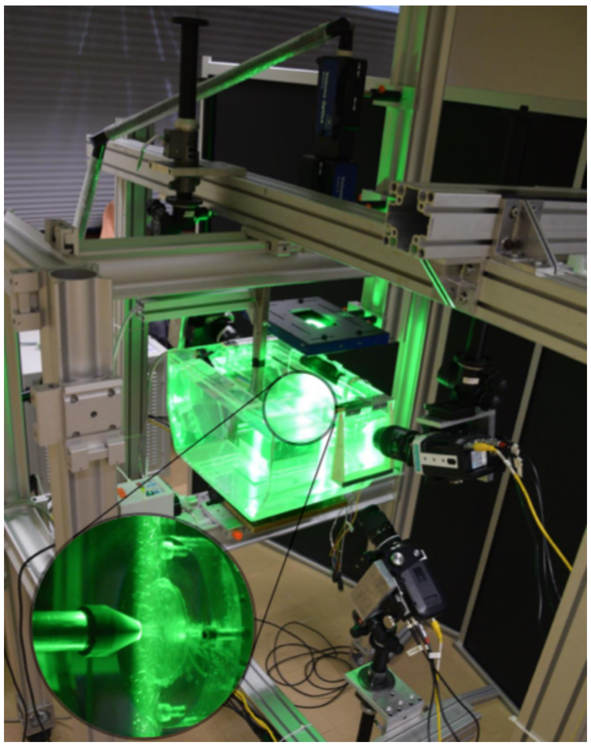
(c)

Fig. 5: Comparison of results produced by KLPT and STB at ppp=0.05 for different image noise levels: (a) mean positional error of detected particles; (b) fraction of undetected particles; (c) fraction of tracked ghost particles.

Fig. 6: Comparison of results produced by KLPT and STB at ppp=0.075 for different image noise levels: (a) mean positional error of detected particles; (b) fraction of undetected particles; (c) fraction of tracked ghost particles.



(a)



(b)

Fig. 7: Experimental arrangement of the volumetric PIV system: (a), Schematic of the volumetric PIV camera configuration; (b), Photograph of the experimental setup.

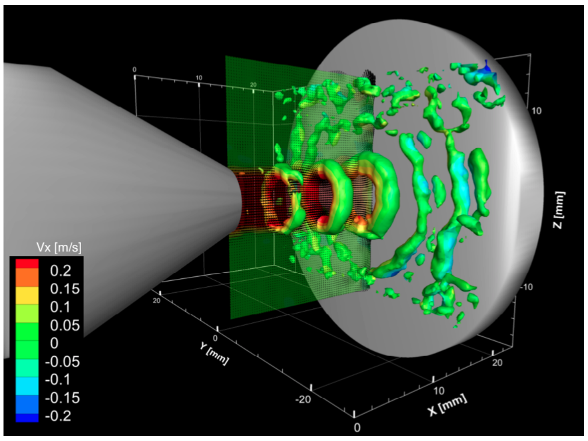


Fig. 8: TomoPIV reconstruction with SMART of the instantaneous wall-normal velocity component superimposed on an isosurface of the Lambda2 vortex criteria, in an impinging jet flow at $Re = 1250$.

code using real data. Real experimental data are characterized by a higher level noises that give rise to large amount of outliers if not properly handled. We implemented an outlier removal procedure that checks both the spatial and temporal coherences.

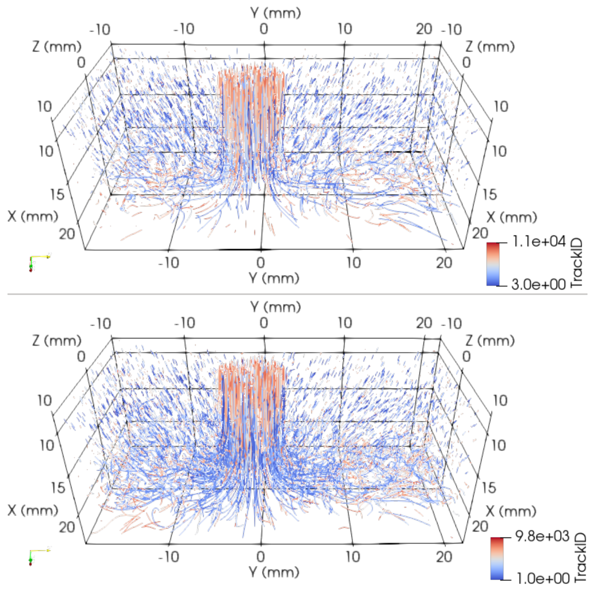
5.2 Tracking evaluation compared to STB

5.2.1 Particle trajectory analysis

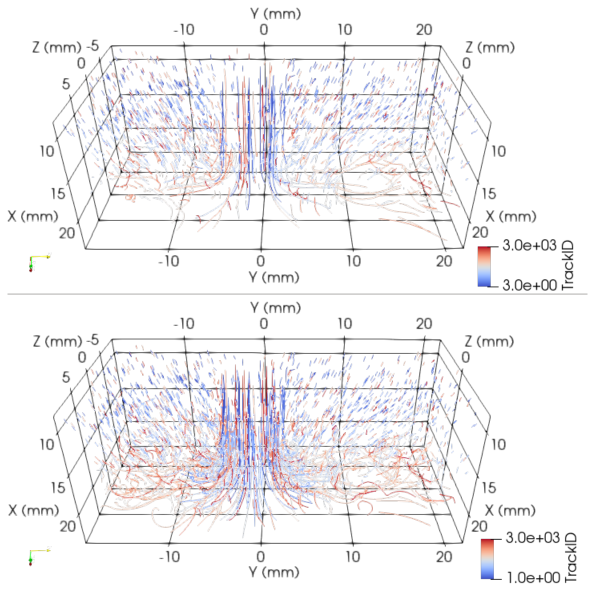
We plot in figure 9 the tracked particle trajectory colored by their ID and observed at frame number 40. Note that only tracks passing through frame 40 are shown. Those tracked particles either terminated before or started after the 40th frame are not displayed. Figure 9a shows all valid tracks. KLPT (9800 tracks) and STB-Davis (11000 tracks) perform equally well at the free jet zone, where the velocity is high, but acceleration is low. They also captured most of the tracks at the ambient region where the particle moves slowly. However, we also observe that KLPT reconstructs more and longer tracks at the impinging region where the acceleration is high. More specifically, at the end of the free jet and at the beginning of the wall jet, the tracks gained by KLPT are dominated by blue color that is reconstructed at an earlier stage. For a clearer view of those tracks, we also plot in figure 9b the trajectories of the first 3000 tracks reconstructed right after the common initialization stage to fully demonstrate the power of KLPT.

We can observe from both figures that for STB, there are practically no blue tracks (reconstructed at an earlier stage) but more red tracks (reconstructed at a later stage) near the impinging region. It indicate that STB Davis is likely to lost track when the particle arrives near the impinging region, although STB can effectively restart the track later. KLPT can build longer tracks and follow them across this highly accelerated impinging region. The same observation is made at the wall jet development region where the turbulence is high. KLPT also outperforms STB significantly both at the impinging region and the wall jet development region. These regions feature complex dynamics that can not be well emulated by statistical filters commonly used in signal processing. Naturally, for STB Davis, the predictor does not work very well, resulting in lost particles followed by diverged optimizations with an inaccurate starting position. Although KLPT employs a simple polynomial-based filter, it can overcome the burdens introduced by an inaccurate starting position as the kernel-based optimization scheme is much more robust.

Now we take a closer look at the impinging and the wall jet development regions. Figure 10 visualizes the first 3000 tracked particles' trajectories at 40th frame, with a similar viewpoint as



(a)



(b)

Fig. 9: Tracked particles at the 40th frame with a tail showing their full trajectories colored by their ID: (a) all tracked particles; (b) a portion of the first 3000 tracked particles found after the initialization stage that still continue at the 40th frame. The left column of each figure is generated from STB Davis, while the right column is generated from KLPT.

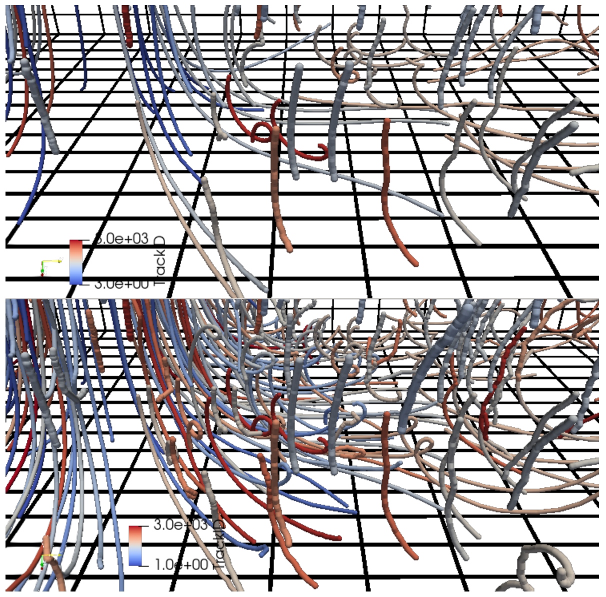


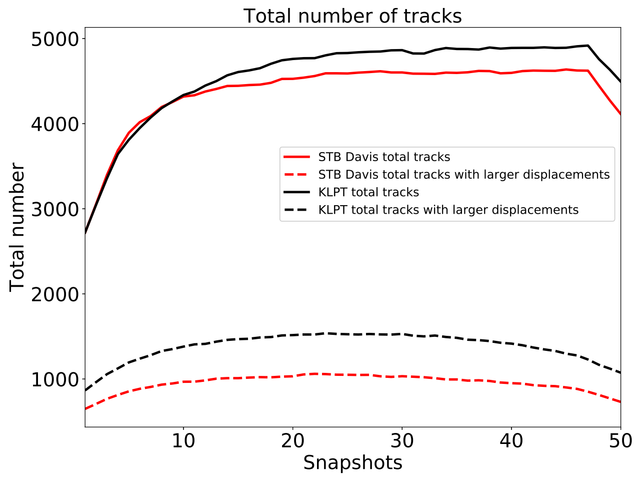
Fig. 10: The detailed view of the first 3000 tracked particles found after the initialization stage still being tracked at the 40th frame with a tail showing their full trajectories colored by their ID: The left figure is generated from STB Davis while the right is generated from KLPT.

in figure 9b, but in a subdomain approximately centered around point $(x = 25mm, y = 5mm)$. We can directly observe that KLPT reconstructs more and longer tracks near the wall. Those tracks belonged to the wall jet. STB Davis has difficulties in following these tracks due to the collision. KLPT also captures the complete structure of a vortex ring across the impinging region and the recirculation region. At the same time, STB Davis can at best reconstruct only segments of the vortex rings.

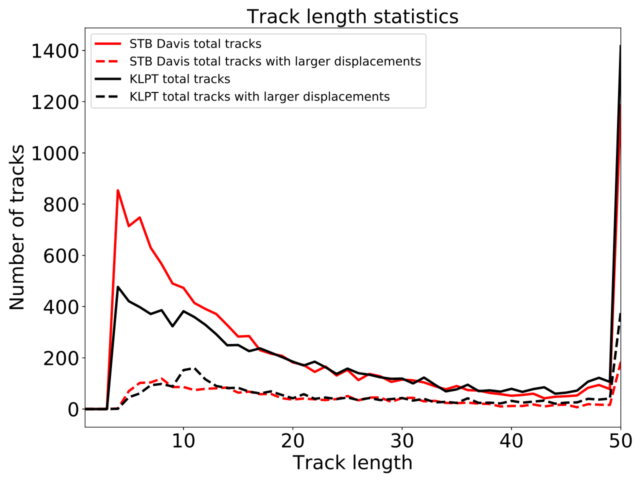
5.2.2 *Track statistics*

We also calculate the track statistics. Figure 11a shows the total number of tracks in terms of the snapshots. Figure 11b shows the distribution of track length for total tracks. The solid curves represent all valid tracks obtained by STB (red) and KLPT (black). In figure 11a we observe that KLPT continuously gained slightly more tracks than STB despite STB yields more tracks in total (11000 versus 9800), which suggests that STB yield more short track segments. This finding is verified in figure 11b where STB has a much higher peak of length distribution at very short tracks (4 to 10). The length distribution of KLPT is flatter and has a higher peak at the end (1417 tracks of length 50 against 1185 for STB).

We notice that many reconstructed tracks are located at the ambient region where the tracking is relatively easy and both methods yield similar results. We want to focus on more interesting flow regions, such as the free jet, the impinging area, and the wall jet. Hence we removed the tracks at the ambient region by considering only those tracks traveling a relative long distance (larger than 3 px). Then we recompute the track statistics in terms of the remaining reconstructed tracks. The results are shown by dotted lines in figure 11. The difference between KLPT and STB, in terms of the number of tracks with larger displacement at each frame, is more considerable than the difference between KLPT and STB in terms of the number of all tracks. It suggests that STB actually yields more segments of tracks. Those short segments potentially belong to a single long track that KLPT can more effectively reconstruct. The track length statistics for both methods become flatter. The first peak length for STB is at 8 time-steps compared 11 time-steps for KLPT. The number of tracks of length 50 of KLPT (381) is also much higher than STB (184).



(a)



(b)

Fig. 11: (a): total number of tracks at each frame. (b): Number of tracks as a function of track length. Solid lines denote the all valid tracks while dotted lines disregard those tracks with small displacement (less than 3 px). STB Davis results are denoted by red lines and KLPT by black lines.

6 1st LPT challenge results

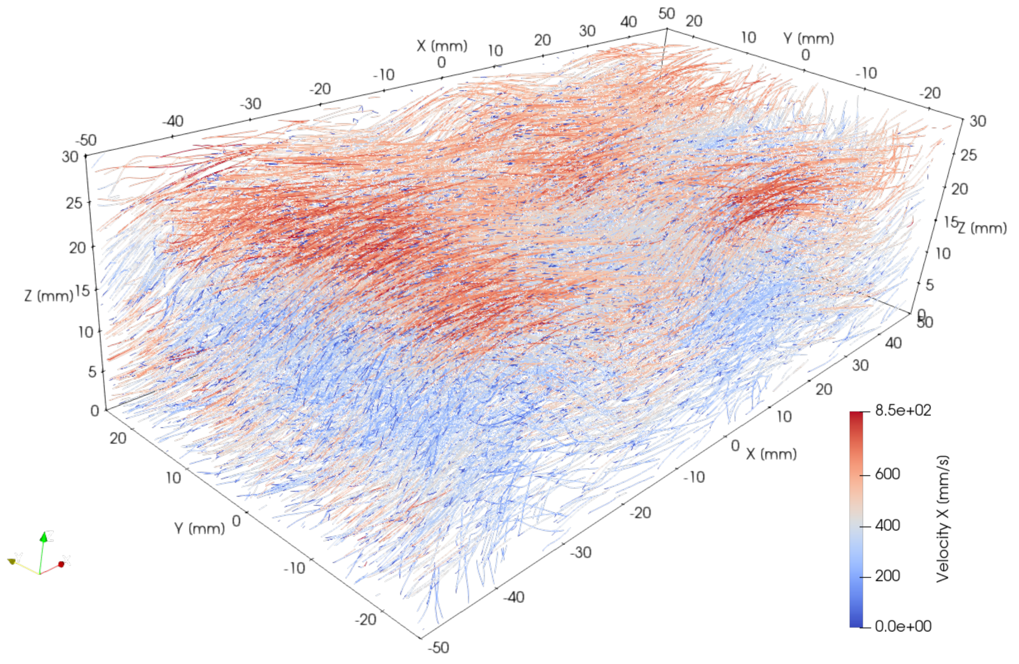
Finally, we demonstrated the results of KLPT w.r.t. the 1st LPT challenge dataset (link). Various seeding density is provided in the challenge data. Figure 12 shows two views of the reconstructed particle tracks for the time-resolved dataset of ppp 0.08, color coded by the stream-wise velocity. Having been verified by the organizers during the 3rd CFD for PIV workshop, our KLPT can yield competitive results state-of-the-art with its state-of-the-art counterparts (prominently STB by LaVision and DLR) until seeding density reaching 0.08 ppp. Beyond that level, our naively implemented IPR fails to provide a reasonably good initial particle field.

7 Conclusions

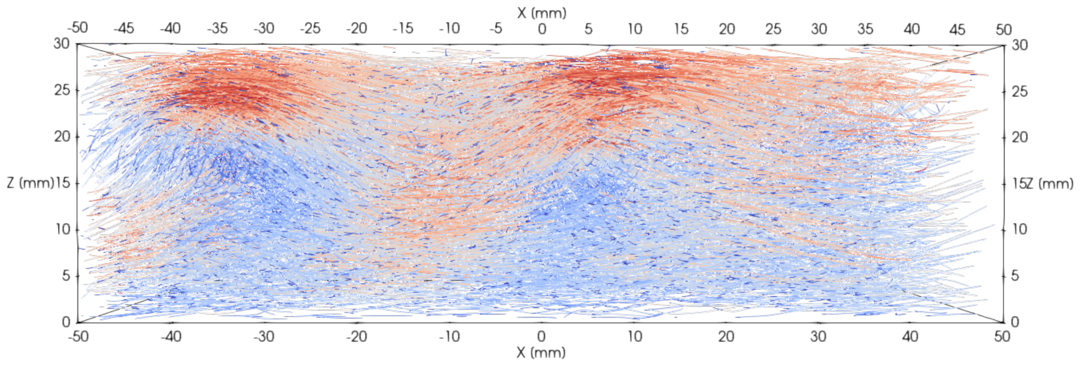
In this article, we have first introduced a novel particle tracking principle, tracking-by-detection. Based on this new principle, we have proposed a new tracking scheme based on function learning, emphasizing kernel function. This new approach is named Kernelized LPT. An important implication of our approach is that it generalizes the comparison operation of two local particle image patches, commonly used in other LPT approaches, to a more powerful kernel function, a higher level more powerful tool measuring the similarities of two input features. The resulting algorithm is both mathematically sound and easy to manipulate.

The proposed method is validated against both synthetic and real datasets. We have achieved significant improvements on both robustness and accuracy compared to STB-in-house and STB in Davis 10. The differences are even more pronounced for high particle density and large time separation. This finding corresponds quite well to our intuition on the weakness of STB: STB only works well when the predictor gives sufficient good results. KLPT is also applied to a real experimental dataset depicting an impinging jet. Compared to the tracks reconstructed by STB in Davis 10, KLPT can yield more tracks and reveal more local flow structure that is likely to be smoothed out by other schemes. We have also tested KLPT on the 1st LPT challenge datasets. Our KLPT has achieved state-of-the-art performance with images of particle density until $\mathrm{ppp} = 0.08$. We also provide an open source library (Particle Tracking Library(PTL)) that features our KLPT algorithm as well as our STB implementations used in this article. We hope that this library serves as a benchmark for our community accelerating the development process for multi-view particle images-related tracking problem, particularly in academic environments.

For future work, we can investigate the use of a more comprehend kernel function and more powerful feature extractor into the KLPT framework in future work. Such investigations can be carried out in two distinct but equally important directions. Firstly, more flow-dependent features based on fluid dynamics can be adopted. For example, Khojasteh et al. (2020) proposed using Lagrangian Coherent Structure (LCS) as an objective criterion to discern whether one particle's motion is coherent or not with its neighborhood. Such classification can lead to a better prediction of the



(a)



(b)

Fig. 12: Two views of the reconstructed particle tracks for images dataset of ppp 0.08, color coded by the stream-wise velocity, containing approximately 100,000 number of tracks.

particle's position. LCS can be viewed as a feature extractor that encodes the interrelations between one particle and its neighboring particles. Based on this notion, more accurate tracking performance can be achieved by considering a new kernel function based on LCS in addition to the dot-product kernel based on local patch's gray intensity. Secondly, as mentioned in section 3.2.2 of this paper, a deep feature extractor based on DNN can be smoothly integrated into the KLPT framework thanks to the kernel trick. Compared to the above more physical feature approach, the deep feature extractor is likely to be more robust, as shown by many studies. However, it remains not easy to interpret the physical meaning of the obtained feature vector. The DNN-related techniques will be discussed in a subsequent paper.

Acknowledgements We would like to thank Anthony Guibert (former INRAE's assistant engineer) who conducted in 2016 these high quality TomoPIV experiments in the impinging jet flow.

References

Álvarez, M. A., Rosasco, L. & Lawrence, N. D. (2012), 'Kernels for Vector-Valued Functions: A Review', *Foundations and Trends® in Machine Learning* **4**(3), 195–266.

Balntas, V., Riba, E., Ponsa, D. & Mikolajczyk, K. (2016), Learning local feature descriptors with triplets and shallow convolutional neural networks, *in* 'Procedings of the British Machine Vision Conference 2016', pp. 119.1–119.11.

Elsinga, G. E., Scarano, F., Wieneke, B. & van Oudheusden, B. W. (2006), 'Tomographic particle image velocimetry', *Experiments in fluids* **41**(6), 933–947.

González, G., Sciacchitano, A. & Scarano, F. (2019), Dense volumetric velocity field reconstruction with time-segment assimilation, *in* 'Proc. 13th Int. Symp. on Particle Image Velocimetry'.

Haasdonk, B. & Bahlmann, C. (2004), Learning with Distance Substitution Kernels, *in*

D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. E. Rasmussen, H. H. Bülthoff, B. Schölkopf & M. A. Giese, eds, 'Pattern Recognition', Vol. 3175, pp. 220–227.

Henriques, J. F., Caseiro, R., Martins, P. & Batista, J. (2014), 'High-speed tracking with kernelized correlation filters', *IEEE transactions on pattern analysis and machine intelligence* **37**(3), 583–596.

Huhn, F., Schanz, D., Manovski, P., Gesemann, S. & Schröder, A. (2018), 'Time-resolved large-scale volumetric pressure fields of an impinging jet from dense Lagrangian particle tracking', *Experiments in Fluids* **59**(5), 81.

Khojasteh, A. R., Heitz, D., Yang, Y. & Laizet, S. (2020), Lagrangian coherent track initialisation, *in* '3rd Workshop and 1st Challenge on Data Assimilation & CFD Processing for PIV and Lagrangian Particle Tracking'.

Maas, H. G., Gruen, A. & Papantoniou, D. (1993), 'Particle tracking velocimetry in three-dimensional flows - Part 1. Photogrammetric determination of particle coordinates', *Experiments in fluids* **15**(2), 133–146.

Malik, N. A., Dracos, T. & Papantoniou, D. A. (1993), 'Particle tracking velocimetry in three-dimensional flows - Part II: Particle tracking', *Experiments in fluids* **15**(4-5), 279–294.

Micchelli, C. A. & Pontil, M. (2005), 'On Learning Vector-Valued Functions.', *Neural Computation* **17**(1), 177–204.

Novara, M., Schanz, D., Geisler, R., Gesemann, S., Voss, C. & Schröder, A. (2019), 'Multi-exposed recordings for 3d lagrangian particle tracking with multi-pulse shake-the-box', *Experiments in Fluids* **60**(3), 44.

Novara, M., Schanz, D., Reuther, N., Kähler, C. J. & Schröder, A. (2016), 'Lagrangian 3D particle tracking in high-speed flows: Shake-The-Box for multi-pulse systems', *Experiments in fluids* **57**(8), 1–20.

Ouellette, N. T., Xu, H. & Bodenschatz, E. (2005), 'A quantitative study of three-dimensional Lagrangian particle tracking algorithms', *Experiments in fluids* **40**(2), 301–313.

Parnaudeau, P., Carlier, J., Heitz, D. & Lamballais, E. (2008), 'Experimental and numerical studies of the flow over a circular cylinder at reynolds number 3900', *Physics of Fluids* **20**(8), 085101.

Scarano, F. (2013), 'Tomographic PIV: principles and practice', *Measurement Science and Technology* **24**(1), 012001–30.

Schanz, D., Gesemann, S. & Schröder, A. (2016), 'Shake-The-Box: Lagrangian particle tracking at high particle image densities', *Experiments in fluids* **57**(5), 1–27.

Schanz, D., Gesemann, S., Schröder, A., Wieneke, B. & Novara, M. (2012), 'Non-uniform optical transfer functions in particle imaging: calibration and application to tomographic reconstruction', *Measurement Science and Technology* **24**(2), 024009.

Schölkopf, B. & Smola, A. J. (2002), *Learning with Kernels*, Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press.

Schröder, A., Willert, C., Schanz, D., Geisler, R., Jahn, T., Gallas, Q. & Leclaire, B. (2020), 'The flow around a surface mounted cube: A characterization by time-resolved PIV, 3D Shake-The-Box and LBM simulation', *Experiments in Fluids* **61**(9), 189.

Shawe-Taylor, J. & Cristianini, N. (2004), *Kernel Methods for Pattern Analysis*, Cambridge University Press.

Sodjavi, K., Brice Montagneé, A., Bragan¸ca, P., Meslem, A., Byrne, P., Degouet, C. & Sobolik, V. (2016), 'PIV and electrodiffusion diagnostics of flow field, wall shear stress and mass transfer beneath three round submerged impinging jets', *Experimental Thermal and Fluid Science* **70**, 417–436.

Soloff, S. M., Adrian, R. J. & Liu, Z.-C. (1999), 'Distortion compensation for generalized stereoscopic particle image velocimetry', *Measurement Science and Technology* **8**(12), 1441–1454.

Steinke, F. & Schölkopf, B. (2008), 'Kernels, regularization and differential equations', *Pattern Recognition* **41**(11), 3271–3286.

Tan, S., Salibindla, A., Masuk, A. U. M. & Ni, R. (2020), 'Introducing OpenLPT: New method of removing ghost particles and high-concentration particle shadow tracking', *Experiments in Fluids* **61**(2), 47.

Tsai, R. (1987), 'A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses', *IEEE Journal on Robotics and Automation* **3**(4), 323–344.

Wieneke, B. (2008), 'Volume self-calibration for 3d particle image velocimetry', *Experiments in Fluids* **45**(4), 549–556.

Wieneke, B. (2012), 'Iterative reconstruction of volumetric particle distribution', *Measurement Science and Technology* **24**(2), 024008–15.

Yang, Y., Heitz, D. & Mémin, E. (2018), An ensemble filter estimation scheme for lagrangian trajectory reconstruction, *in* 'Congrès Francophone de Techniques Laser, CFTL 2018'.

Yang, Y., Robinson, C., Heitz, D. & Mémin, E. (2015), 'Enhanced ensemble-based 4dvar scheme for data assimilation', *Computers and Fluids* **115**(C), 201–210.

Zabih, R. & Woodfill, J. (1994), Non-parametric local transforms for computing visual correspondence, *in* 'European conference on computer vision', Springer, pp. 151–158.

A Dynamics as regularizer to smooth trajectory

Following the discussion in section 3.2.4, here we show how to obtain an explicit form of regularizer $\mathbf{R}$ from dynamical constraints. Given l frames from k to $k+l-1$, we can formulate the propagation of particle p as: $\mathbf{X}_p^{k+1} = \mathbf{M}\mathbf{X}_p^k + \boldsymbol{\xi}_p^{k+1}$ where we use instead an linear operator $\mathbf{M}$ for sake of clarity. Then the term $||\mathbf{Rf}||^2$ can be expressed in terms of the trajectory of particle p over the frames:

$$\begin{pmatrix} \mathbf{B}^{-\frac{1}{2}} & & & \\ & \mathbf{Q}_1^{-\frac{1}{2}} & & \\ & & \ddots & \\ & & & \mathbf{Q}_{l-1}^{-\frac{1}{2}} \end{pmatrix} \begin{pmatrix} \mathbf{I} & & & \\ -\mathbf{M} & \mathbf{I} & & \\ & \ddots & \ddots & \\ & & -\mathbf{M} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \delta\mathbf{X}_0 \\ \delta\mathbf{X}_1 \\ \vdots \\ \delta\mathbf{X}_{l-1} \end{pmatrix} = \begin{pmatrix} \sigma_0 \\ \xi_1 \\ \vdots \\ \xi_{l-1} \end{pmatrix}. \tag{13}$$

The increments are defined in following:

$$\delta\mathbf{X}_0 = \mathbf{X}_0 - \mathbf{X}_b \tag{14}$$

$$\delta\mathbf{X}_1 = \mathbf{X}_1 - \mathbf{M}_{0,1}\mathbf{X}_b \tag{15}$$

$$\vdots$$

$$\delta\mathbf{X}_l = \mathbf{X}_l - \mathbf{M}_{0,l}\mathbf{X}_b \tag{16}$$

Note that $\sigma_0 \sim \mathbb{N}(0, \mathbf{B})$ denotes the uncertainties associated with the prior state $\mathbf{X}_b$. We can identify

$$\mathbf{R} = \begin{pmatrix} \mathbf{B}^{-\frac{1}{2}} & & & \\ & \mathbf{Q}_1^{-\frac{1}{2}} & & \\ & & \ddots & \\ & & & \mathbf{Q}_{l-1}^{-\frac{1}{2}} \end{pmatrix} \begin{pmatrix} \mathbf{I} & & & \\ -\mathbf{M} & \mathbf{I} & & \\ & \ddots & \ddots & \\ & & -\mathbf{M} & \mathbf{I} \end{pmatrix} \tag{17}$$

Given $\mathbf{R}$, we can recover the Gram matrix as well as the kernel function. Eventually, we can have a similar solution form as (11) that resolves the particle trajectory simultaneously by leveraging its temporal coherence and the sequential image data. This kind of solution provides a smoothing effect because the state $\mathbf{X}_k$ at frame k is optimally conditioned on the following data $\mathbb{I}$ available after frame k.

B Equivalence between EnDA solution (12) and KLPT solution (11)

First we consider in definition (9), $\kappa(\mathbb{I}^i, \mathbb{I}^j) = (\mathbb{I}^j)^T\mathbb{I}^i$ is the scalar dot-product kernel and $\mathbf{C}$ is an identity matrix $\mathbf{I}_{dd} \in \mathbb{R}^{d\times d}$ indicating uncorrelated spatial field. Then the ith component of $\mathbf{X}_p^{'}$ can be derived from (11) with centered data pairs:

$$(\mathbf{X}_p^{'})_i = [\kappa(\mathbb{I}_p^{'1}, \mathbb{I}_{p*}^{res})\mathbf{I}_{dd}, \ldots, \kappa(\mathbb{I}_p^{'1}, \mathbb{I}_{p*}^{res})\mathbf{I}_{dd}]_i \begin{bmatrix} (\kappa_{11}+\lambda)\mathbf{I}_{dd} & & \kappa_{1N}\mathbf{I}_{dd} \\ & \ddots & \\ \kappa_{N1}\mathbf{I}_{dd} & & (\kappa_{NN}+\lambda)\mathbf{I}_{dd} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{X}_p^{'1} \\ \vdots \\ \mathbf{X}_p^{'N} \end{bmatrix} \tag{18}$$

It can be easily proven that the inverse of the matrix $\overline{\overline{K}} + \lambda\mathbf{I}_{dN}$ maintains the same structure. The coefficient of each block equals to the corresponding scalar version of Gram matrix ($N \times N$) as following derived from formulation (12)

$$(\mathbf{X}_p^{'})_i = [\mathbf{X}_p^{'1}, \cdots, \mathbf{X}_p^{'N}]_i \begin{bmatrix} \kappa_{11}+\lambda & & \kappa_{1N} \\ & \ddots & \\ \kappa_{N1} & & \kappa_{NN}+\lambda \end{bmatrix}^{-1} \begin{bmatrix} \kappa(\mathbb{I}_p^{'1}, \mathbb{I}_{p*}^{res}) \\ \vdots \\ \kappa(\mathbb{I}_p^{'N}, \mathbb{I}_{p*}^{res}) \end{bmatrix} \tag{19}$$

Then with some basic matrix multiplication, it can be shown that those two formulations indeed yield the same result.