SoftwareX 11 (2020) 100391

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

μDIC: An open-source toolkit for digital image correlation

Sindre Nordmark Olufsen [a,*], Marius Endre Andersen [b], Egil Fagerholt [a]

[a] *Structural Impact Laboratory (SIMLab), Department of Structural Engineering, NTNU, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway*
[b] *SINTEF Industry, Department of Materials and Nanotechnology, NO-7034 Trondheim, Norway*

A R T I C L E I N F O

Article history:
Received 29 May 2019
Received in revised form 25 October 2019
Accepted 17 December 2019

Keywords:
Digital Image Correlation
B-splines
Speckle image generation
Virtual experiments
Python

A B S T R A C T

We here present a Digital Image Correlation toolkit, formulated as a Python package. This package aims at providing a complete toolkit for performing DIC analysis on experimental data, performing virtual experiments, as well as a framework for further development. A suite of tools for generating synthetic speckle images, modelling of sensor artefacts and deformation of images by displacement fields, are included. The virtual experiments are used as a part of the accuracy assessment of the toolkit as well as for testing during development. B-spline elements are employed for the discretisation of the displacement fields and allow the polynomial order and degree of continuity to be controlled by the user.

Code metadata

Current code version	0.1.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2019_193
Legal Code License	MIT
Code versioning system used	GIT
Software code languages, tools, and services used	Python, Numpy, Scipy, Matplotlib, Numba
Compilation requirements, operating environments &	Cross platform
If available Link to developer documentation/manual	https://mudic.readthedocs.io/en/latest/
Support email for questions	sindre.n.olufsen@ntnu.no

1. Motivation and significance

Digital Image Correlation (DIC) has become a popular tool for quantification of surface deformations, due to its ease of use and ability to provide local deformation measurements even for inhomogeneous deformation fields [1,2]. DIC is performed by processing a series of images captured during the deformation of the object of interest, and a mapping between the image coordinates of an image of the undeformed (reference) object and the image coordinates of an image of the deformed object is calculated. This operation is often facilitated by applying a speckle-like pattern on the surface of the specimen. In order to obtain such a mapping, an optimisation scheme is employed, searching for the mapping which gives the best correlation between a reference image and the current image. The coordinate mapping is in turn is used to calculate the strain fields.

DIC has been a valuable tool in a vast range of scientific fields such as characterisation of biological materials [3–6], detection of tumours [7], nanoscale investigation of materials using TEM [8], characterisation of mechanical properties of polymers [9–11], and many others.

There are several variants of DIC, the most common being subsets-based DIC (local DIC) and Finite-Element-based DIC (global DIC). Subset-based DIC attempts to correlate independent subsets of the image, often being in the order of ten to thirty pixels in each direction. In order to cover a large portion of the specimen surface, a large number of subsets are used. FE-based DIC, on the other hand, discretises the deformation field using finite elements, approximating the inter-nodal coordinates by interpolation. FE-based DIC normally enforces displacement continuity, but by using higher order element formulations [12, 13], B-splines [14] or NURBS [15], higher order continuity can be enforced. Interestingly, a performance comparison of subset- and FE-based DIC was done by Hild et al. [16] and later by Pan et al. [17], reaching opposing conclusions. Recent advances in the

* Corresponding author.
E-mail address: sindre.n.olufsen@ntnu.no (S.N. Olufsen).

field of DIC include self-adaptive element order [18] and more computationally efficient formulations [19].

Commercial DIC software such as ISI-Sys VIC and GOM-correlate, and open source DIC codes such as DICE [20] and nCorr [21] are now available. For scientific use, it can be appealing to use open source software, as insight into the internal architecture of the software is available. However, at the time of writing, the available open source DIC codes were either formulated in C++, calling for significant programming knowledge, bound to commercial software such as MATLAB, or formulated as scripts with very limited testing and documentation.

This project aims to provide a complete and easy to use toolkit for 2D digital image correlation, that is verified and provided with automated testing, implemented in an accessible programming language. Python was chosen for this task, aided by NumPy [22], SciPy [23] and Numba [24] for fast array operations. Visualisation is provided by Matplotlib [25].

Motivated by the improved accuracy and flexibility reported by Cheng et al. [14], B-splines are used to discretise the deformation field. B-splines allows for a very compact and flexible formulation where the user can specify the degree of the interpolation polynomials and the degree of continuity as most appropriate for their application. Linear and parabolic element formulations such as Q4 and Q8 are subsets of the B-splines and are therefore an integrated part of the package.

A modified Newton–Raphson iterative solver [12] is used to minimise the sum of squared differences between the reference and current image. During the optimisation process, the inter-pixel values of the image are obtained by interpolation using bi-cubic or bi-quintic splines.

An additional but important component is a toolkit for generating and deforming speckle images, including camera artefacts such as fill-factor and noise. This toolkit is based on the ideas presented by Orteu et al. [26]. The toolkit allows for assessment of element formulations, aliasing artefacts and filtering, and is an integral part of the provided suite of integration tests. Additional validation is provided by experimental data where other techniques have been used to measure deformations.

The user can implement this package in the daily workflow by adopting one of the provided example-scripts, where in most cases the only required alteration is the path to the folder containing the images to be analysed.

This toolkit was conceived as a Matlab-code during the PhD study of Dr Andersen [12] and has been used in several studies [10–12]. It was later rewritten in Python by the corresponding author and has been applied in [9,27].

2. Software description

The μDIC toolkit provides a collection of high-level functions and classes, used for importing images, generating synthetic input data, meshing, image correlation and post-processing. The API is constructed such that the functions and classes can be used as stand-alone components. Factories are included in all packages, providing default behaviour for the most common tasks. The toolkit can be used in the daily workflow when performing real experiments or used for exploring the impact of different factors such as element formulations, speckle design and filtering by performing virtual experiments. An interactive experience can be obtained by using IPython [28] or Jupyter notebooks. Testing is an integral part of this toolkit, and virtual experiments, simulating know deformation fields, are used as integration tests along with other unit tests.

2.1. Software architecture

The μDIC tookit is formulated as a set of packages:

- IO: Tools for image import and export
- VirtualLab: Tools for synthetic data generation
- Mesher: Tools for mesh generation and manipulation
- Correlator: The digital image correlation routines
- PostProcessor: Tools for post-processing and visualisation

The packages can be combined, providing a complete processing pipeline tailored to the user's needs. A typical pipeline and the most relevant package components are shown in Fig. 1 and run as follows:

An *ImageStack* object is generated either by loading a set of images by using *imagestack_from_folder* or by constructing a *VirtualExperiment*. A *Mesher* is instantiated and a *mesh* object is obtained by passing the *ImageStack* to the *mesh* method. The resulting *Mesh* object and *ImageStack* is then stored in the *DICInput* object. The *DICAnalysis* factory verifies *DICInput* and a *DICJob* is constructed. The DIC analysis can now be initiated by calling the *run* method. The resulting *DICRes* object contains all relevant output from the DIC analysis, and field variables can be calculated by *DICPost* class and visualised by the *plot* function.

2.2. Software functionality

The core functionality of the main packages in the toolkit is presented in the following.

2.2.1. IO

The IO package provides all the tools needed for importing images and saving objects. The most typical tasks, such as making an image stack of all images in a folder with a given extension, are made easier by providing factories such as *imagestack_from_folder*.

2.2.2. Virtual lab

The VirtualLab package provides tools for the generation of synthetic speckle images, deformation of speckle images using prescribed deformation fields and modelling of camera artefacts such as fill-factor and pixel eccentricity. The virtual lab toolkit is based on the work of Orteu et al. [26]. The main modules of the package and their intended usage are illustrated in Fig. 2 and described below:

- The Speckle module contains tools for generating spray-paint-like speckles using a Perlin noise [29] based algorithm or the Rosta algorithm found in Appendix. Tools for generating speckles formed by circular dots or bi-harmonic fields are also included. The resolution, speckle size and degree of smoothing can be controlled by the user.
- The Deformer module contains tools for distorting the speckle images according to a prescribed displacement field or by a deformation gradient. Inversion of the displacement function is done by employing a Newton solver. Bi-quintic splines are used to interpolate the grey-scale values of the image.
- The Downsampler module contains tools for down-sampling of images, including sensor artefacts such as fill-factor and pixel eccentricity. The sensor is modelled as a grid where each pixel of the sensor covers a finite region of the image. The grey scale values of the image covered by a pixel of the sensor are then integrated, giving the grey scale value of the sensor pixel. Each pixel on the sensor can be offset by a random value picked from a Gaussian distribution with a prescribed standard deviation. The position of the super-sampled points covered by each pixel of the sensor is calculated, as shown in [26].

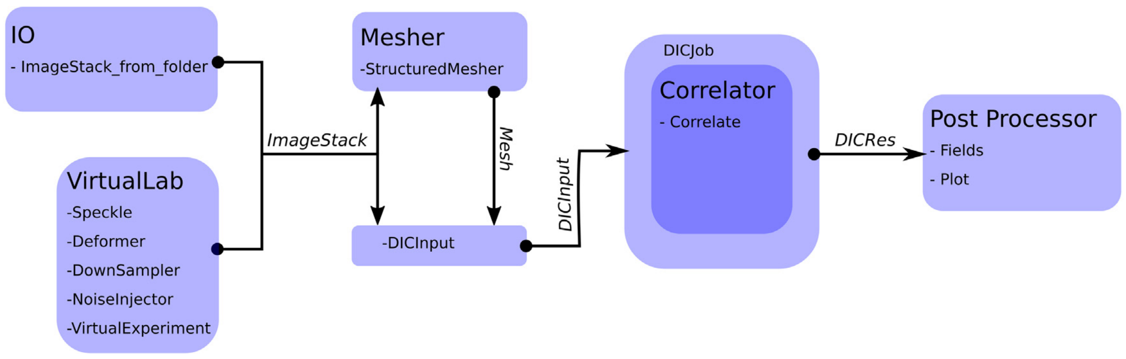

Fig. 1. An example pipeline showing the main packages and the most relevant components.

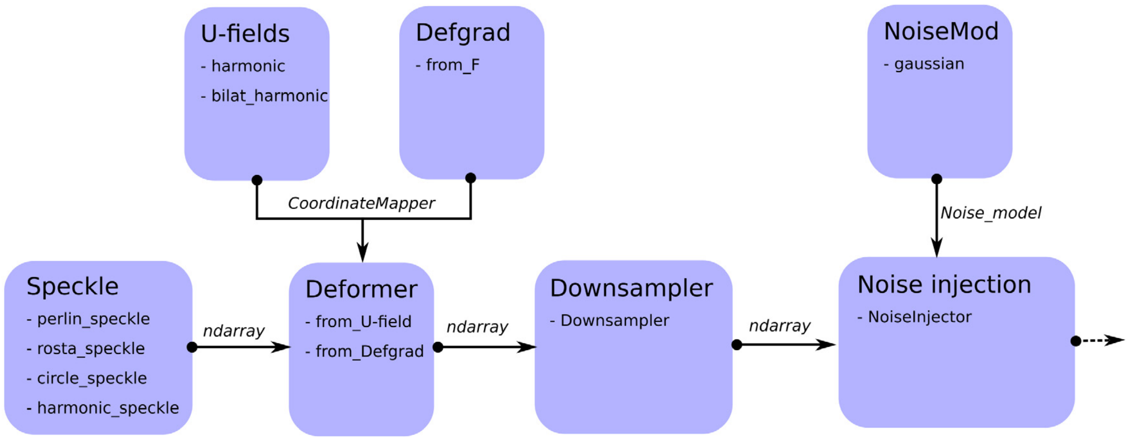

Fig. 2. Typical use of VirtualLab to create speckle images with a given deformation and noise level. Module names and constructors are shown.

- The NoiseInjector module is used to introduce noise to the downsampled images. The user can define a noise model and the standard deviation of the noise distribution. An additive Gaussian noise model is provided with the toolkit, and the user can easily add other models.

2.2.3. Mesher

The Mesher package contains the tools for mesh generation and manipulation. B-splines are used to discretise the deformation field into a bi-directional grid of k x l control points. The formulation and algorithms are taken from [30]. The coordinates $\mathbf{x}(u, v)$ on the B-spline surface is given by:

$$\mathbf{x}(u, v) = \sum_{i=0}^{k} \sum_{j=0}^{l} N_{i,p}(u) N_{j,q}(v) \mathbf{m}_{ij} \tag{1}$$

where $N_{i,p}$ and $N_{j,q}$ are the B-spline basis functions, u and v are the spline coordinates in the range [0, 1] and p and q denotes the polynomial order. $\mathbf{m}_{ij}$ contains the coordinates of the control points along the bi-directional grid axes of the spline surface. By rearranging, Eq. (1) can be written on the form:

$$\mathbf{x}(u, v) = \boldsymbol{A}(u, v)\mathbf{n} \tag{2}$$

where $\mathbf{n}$ is a vector contains all the control point positions and $\boldsymbol{A}(u, v)$ is a matrix containing the values of the B-spline basis functions.

2.2.4. Correlator

The Correlator package contains the Digital Image Correlation routines. A modified Newton scheme is used to minimise the sum of squared differences between the grey scale values of the reference image and the current image. The derivation of the correlation routine follows [31] except for the chosen correlation criterion.

The objective of the correlation routine is to find the current image coordinates for each pixel $\boldsymbol{x}$ such that:

$$I_C(\mathbf{x}) = I_R(\mathbf{x}_0) \tag{3}$$

where $I_C(\mathbf{x})$ is the grey-scale intensities of the current image at the pixel coordinates $\mathbf{x}$ and $I_R(\mathbf{x}_0)$ is the grey-scale intensities of the reference image at the initial pixel coordinates $\mathbf{x}_0$. In real world conditions, Eq. (3) cannot be satisfied directly, and it is therefore approximated by using the sum of squared differences (SSD) between the grey scale values of the current and reference image as correlation criterion. By using Eq. (2), the SSD correlation criterion can be written in terms of control point positions as:

$$R = \|\bar{I}_C(\mathbf{n}) - \bar{I}_R(\mathbf{n}_0)\|_2 \tag{4}$$

where $\mathbf{n}_0$ denotes the initial control point position. The approximate solution of Eq. (3) is then found as:

$$\min_{\mathbf{n}} R \tag{5}$$

In order to find the control point positions $\boldsymbol{n}$ which satisfies Eq. (5), a Newton Raphson scheme is used:

$$\nabla\nabla R(\mathbf{n}_i)[\mathbf{n}_{i+1} - \mathbf{n}_i] = -\nabla R(\mathbf{n}_i) \tag{6}$$

where i denotes the increment. The term $\nabla\nabla R(\mathbf{n_i})$ is simplified by assuming that $\bar{I}_C(\mathbf{n}) \approx \bar{I}_R(\mathbf{n}_0)$ as proposed by [31] and by using the grey scale gradients of the reference image instead of the current image, giving:

$$\mathbf{n}_{i+1} = \mathbf{n}_i + (\boldsymbol{b}^T\boldsymbol{b})^{-1}\boldsymbol{b}^T \Delta\bar{\bar{\boldsymbol{I}}}_i \tag{7}$$

where

$$\boldsymbol{b} = \frac{\partial I_R}{\partial \mathbf{x}} \boldsymbol{A} \tag{8}$$

Here, $\Delta \bar{\boldsymbol{I}}_i$ is the difference in grey-scale value between the current and reference frame.

The inter-pixel values of $\bar{I}_C(\mathbf{n}_i)$ and $\bar{I}_R(\mathbf{n}_0)$ are determined by means of bi-cubic or bi-quintic spline interpolation, which has been demonstrated to minimise the bias error [32]. For details and discussion regarding the derivation of the correlation routine, see [31] and [12].

2.2.5. Post processing

Based on the B-spline surface $\mathbf{x}_t(u, v)$ at frame t, the deformation gradient for each material point within the surface can be calculated as [33]:

$$\mathbf{F}_t = \begin{vmatrix} x_{t,u} & x_{t,v} \\ y_{t,u} & y_{t,v} \end{vmatrix} \begin{vmatrix} x_{0,u} & y_{0,v} \\ x_{0,u} & y_{0,v} \end{vmatrix}^{-1} \tag{9}$$

where the entries of $\mathbf{F}_t$ are partial derivatives of the spatial coordinate component fields with respect to u and v. Based on the resulting deformation gradient $\mathbf{F}_t$, the right Cauchy–Green tensor is calculated:

$$\mathbf{C}_t = \mathbf{F}_t^T \mathbf{F}_t = \mathbf{U}_t^2 \tag{10}$$

$\mathbf{U}_t$ is found by calculating the square root of the eigenvalues of $\mathbf{U}_t^2$ as found by spectral decomposition. The Hencky strain tensor $\boldsymbol{\varepsilon}_t$ (often referred to as true or logarithmic strain) is then determined as:

$$\boldsymbol{\varepsilon}_t = \ln(\mathbf{U}_t) \tag{11}$$

Note that the eigenvectors of $\boldsymbol{\varepsilon}_t$ and $\mathbf{U}_t^2$ are now coaxial. The Hencky strain tensor $\boldsymbol{\varepsilon}_i$ is then rotated into the coordinate frame of the image, giving the Hencky strain components in image coordinates.

3. Illustrative examples

3.1. Example 1: Processing experimental data

Processing of experimental data is one of the routine tasks of a scientist. We here show an example where Digital Image Correlation is performed on images that were acquired during a tensile test of a flat dog-bone specimen made of Docol 600L. The output of the procedure is the longitudinal true strain field shown in Fig. 3.

The complete script used to produce these results is shown below:

```
import muDIC as dic

path = r'./example_data/'
images = dic.IO.image_stack_from_folder(path,
 ↪  file_type='.tif')
mesher = dic.Mesher(deg_e=3, deg_n=3)
mesh = mesher.mesh(images)

settings = dic.DICInput(mesh, images)
job = dic.DICAnalysis(settings)
dic_results = job.run()

fields = dic.Fields(dic_results)
viz = dic.Visualizer(fields,images=images)
viz.show(field="true strain", component = (1,1),
 ↪  frame = 230)
```

3.2. Example 2: Performing a virtual experiment

When the user wants to evaluate the performance of the digital image correlation routine or assess the influence of parameters such as speckle design and element formulations, the VirtualLab module can be used. We here show an example where a speckle with a resolution of 16 Mpx is generated using the Rosta algorithm and deformed by a bi-harmonic displacement field. After deformation, the speckle is down-sampled by a factor of 8 and noise is added. DIC is then performed on the generated data set. The code for performing this is shown below:

```
import muDIC as dic
import muDIC.virtualLab as vlab

down_sampling_factor = 8
img_shape = (4000,4000)

speckle_image =
 ↪  vlab.rosta_speckle(img_shape,dot_size=3,
 ↪  density=0.6, smoothness=2.0)
displacement_function =
 ↪  vlab.deformation_fields.harmonic_bilat
image_deformer =
 ↪  vlab.imageDeformer_from_uFunc(
 ↪  displacement_function, omega=3 * np.pi,
 ↪  amp=2.0)
downsampler = vlab.Downsampler(image_shape=
 ↪  super_image_shape,
 ↪  factor=down_sampling_factor, fill=.95,
 ↪  pixel_offset_stddev=0.05)
noise_injector =
 ↪  vlab.noise.noise_injector("gaussian",
 ↪  sigma=.02)
deformed_speckle_generator =
 ↪  vlab.VirtualExperiment(speckle_image,
 ↪  image_deformer, downsampler,
 ↪  noise_injector, n=2)
image_stack =
 ↪  dic.IO.ImageStack(deformed_speckle_generator)

mesher = dic.Mesher(deg_e=3, deg_n=3)
mesh = mesher.mesh(image_stack)

settings = dic.DICInput(mesh, image_stack)
job = dic.DICAnalysis(settings)
dic_results = job.run_analysis()
```

The x-component of the imposed displacement field and the deviation between the DIC results and the imposed field is shown in Fig. 4, demonstrating how the accuracy of the DIC routines can be assessed.

4. Impact

Digital Image Correlation has been widely adopted in several fields of science due to its practicality in real-world experimental setups as well as the amount of detail which can be obtained from the measurements. The current tool aims to provide an easy-to-use, transparent and powerful toolkit for researchers and industry. The toolkit can be applied to a broad range of research problems where quantification of surface deformations is of interest. The popularity of Python makes it an ideal language for such a tool, motivating adoption, modification and extension. By distributing the toolkit via the Python package manager along with the unit test, and providing the user with examples and documentation, the threshold for taking the toolkit into use is lowered. The authors would like to highlight the following:

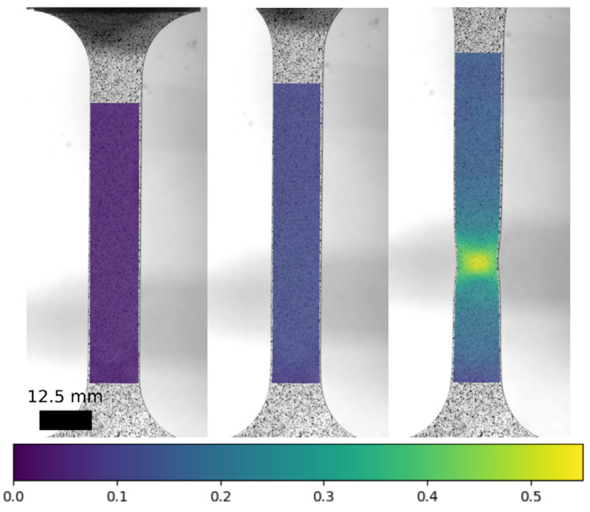

Fig. 3. Results from the digital image correlation pipeline showing the longitudinal true strain for a deformed Docol 600L dog-bone specimen at three stages of deformation.

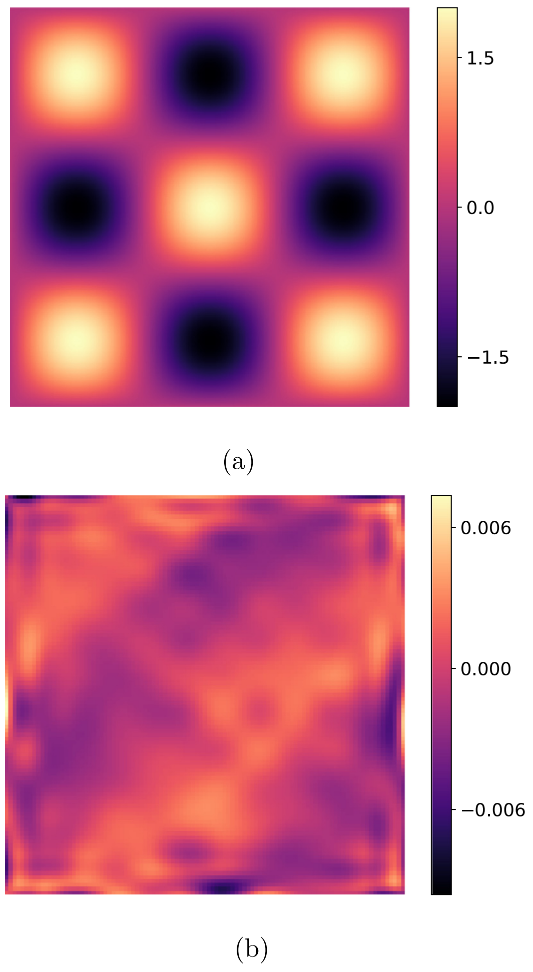

Fig. 4. Results from the virtual experiment. The x-component of the imposed displacement field is shown in figure (a) and the deviation between the DIC results and the imposed field is shown in (b).

- B-splines discretisation of the deformation field provides a flexible formulation where the user can control the degree of the polynomials and the order of continuity.
- Evaluation of the internals such as the correlation routines, element formulations and interpolation routines, is here facilitated through the use of the VirtualLab module.
- Due to the use of the VirtualLab module in the integration tests, modification and extension of the toolkit is made robust, as the impact of the modification is easily evaluated.
- As the toolkit is made available as a Python package, all data processing can be done without changing to other systems. This motivates the adoption of the vast ecosystem of Python packages for tasks such as visualisation, optimisation and machine learning.
- Large sets of experimental data are easily processed by scripting, and example scripts for this task are provided along with the package.
- As the toolkit is distributed under the MIT license, commercial adoption of the package and derivatives of the package are facilitated.

5. Conclusions

We here present a toolkit for Digital Image Correlation and outline its capabilities. The toolkit includes tools for synthetic input generation, a key feature for validation and further development. B-spline discretisation of the deformation field is used, providing a flexible framework where the degree of the polynomials and the order continuity can be specified. Python is chosen for the implementation of the toolkit, making it accessible to a broader user group and motivating use and modification.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors gratefully appreciate the financial support from the Research Council of Norway through the Centre for Advanced Structural Analysis, Project No. 237885 (SFI-CASA).

Appendix. The Rosta algorithm

The following thought experiment formed this algorithm: If a haze of droplets from a spray nozzle travel through unsteady air and land on a surface, they may consolidate into larger droplets. The probability for a droplet to consolidate with another droplet should be coupled to the density of droplets in close proximity. If the density of droplets is higher than some threshold value, the droplets are anticipated to interact and to form a cluster. Based on this thought experiment, the following algorithm is proposed:

A comparison of a subsection of a speckle generated by the algorithm and a real speckle is shown in Fig. A.5.

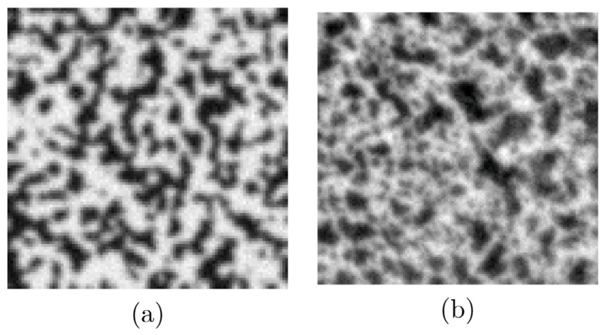

Fig. A.5. A comparison of (a) a synthetic speckle and (b) a real speckle.

Algorithm 1: The Rosta algorithm

```
Input: $s$: an integer representing the image size;
$\sigma_p$: the size of the proximity kernel;
$\sigma_s$: the size of the smoothing kernel;
$n$: the number of repetitions  $t$: fill fraction
Output: $I$: speckle image
make an empty image, $I$ with size $s$ x $s$ ;
make gaussian kernel $k_p$ with $\sigma = \sigma_p * s$;
make gaussian kernel $k_s$ with $\sigma = \sigma_s * s$;
for  $n$ layers do
    make random number field, $f$ of size $s$;
    calculate $g = f * k_p$;
    scale $g$ to unity ;
    find $v$ such that a fraction of t pixels are found by $g < v$
    move values of $g < v$ into $I$;
end
calculate $I = I * k_s$ ;
```

References

[1] Sutton Michael A, Orteu Jean-José, Schreier Hubert W. Image Correlation for Shape, Motion and Deformation Measurements, Vol. 53. Springer Science+Business Media; 2009, p. 1689–99, 9.

[2] Pan Bing, Qian Kemao, Xie Huimin, Asundi Anand. Two-dimensional digital image correlation for in-plane displacement and strain measurement: A review. Meas Sci Technol 2009;20(6).

[3] Zhang Dongsheng, Arola Dwayne D. Applications of digital image correlation to biological tissues. J Biomed Opt 2004;9(4):691.

[4] Rizzuto E, Carosio S, Del Prete Z. Characterization of a digital image correlation system for dynamic strain measurements of small biological tissues. Exp Tech 2016;40(2):743–53.

[5] Rizzuto Emanuele, Carosio Silvia, Musaro Antonio, Del Prete Zaccaria. A Digital Image Correlation based technique to control the development of a skeletal muscle engineered tissue by measuring its surface strain field. In: 2015 IEEE International Symposium on Medical Measurements and Applications, MeMeA 2015 - Proceedings. 2015, p. 314–8.

[6] Palanca Marco, Tozzi Gianluca, Cristofolini Luca. The use of digital image correlation in the biomechanical area: A review. Int Biomech 2016;3(1):1–21.

[7] Krehbiel JD, Lambros J, Viator JA, Sottos NR. Digital image correlation for improved detection of basal cell carcinoma. Exp Mech 2010;50(6):813–24.

[8] Liu Yang, Xia Shuman, Mao Scott X, Pan Zhipeng, Wang Xueju, Zhu Ting, Wang Jiangwei, Fan Feifei. Nanoscale deformation analysis with high-resolution transmission electron microscopy and digital image correlation. J Appl Mech 2015;82(12):121001.

[9] Olufsen Sindre, Clausen Arild Holm, Hopperstad Odd Sture. Influence of stress triaxiality and strain rate on stress-strain behaviour and dilation of mineral-filled PVC. Polym Test 2019;75:350–7.

[10] Johnsen Joakim, Grytten Frode, Hopperstad Odd Sture, Clausen Arild Holm. Influence of strain rate and temperature on the mechanical behaviour of rubber-modified polypropylene and cross-linked polyethylene. Mech Mater 2017;114:40–56.

[11] Johnsen Joakim, Grytten Frode, Hopperstad Odd Sture, Clausen Arild Holm. Experimental set-up for determination of the large-strain tensile behaviour of polymers at low temperatures. Polym Test 2016;53:305–13.

[12] Andersen Marius. An Experimental and Numerical Study of Thermoplastics at Large Deformations (PhD thesis), 2016.

[13] Ma Shaopeng, Zhao Zilong, Wang Xian. Mesh-based digital image correlation method using higher order isoparametric elements. J Strain Anal Eng Des 2012;47(3):163–75.

[14] Cheng Peng, Sutton Michael A, Schreier Hubert W, McNeill Stephen R. Full-field speckle pattern image correlation with B-spline deformation function. Exp Mech 2002;42(3):344–52.

[15] Réthoré J, Elguedj T, Simon P, Coret M. On the use of NURBS functions for displacement derivatives measurement by digital image correlation. Exp Mech 2010;50(7):1099–116.

[16] Hild F, Roux S. Comparison of local and global approaches to digital image correlation. Exp Mech 2012;52(9):1503–19.

[17] Wang Bo, Pan Bing. Subset-based local vs. finite element-based global digital image correlation: A comparison study. Theor Appl Mech Lett 2016;6(5):200–8.

[18] Wittevrongel L, Lava P, Lomov SV, Debruyne D. A self adaptive global digital image correlation algorithm. Exp Mech 2015;55(2):361–78.

[19] Pan B, Li K, Tong W. Fast, robust and accurate digital image correlation calculation without redundant computations. Exp Mech 2013;53(7):1277–89.

[20] Turner Dan, Crozier Paul, Reu Phil. Digital image correlation engine, version 00. 2015.

[21] Blaber J, Adair B, Antoniou A. Ncorr : Open-source 2d digital image correlation matlab software. 2015.

[22] Oliphant Travis E. Guide to NumPy. 2nd ed.. USA: CreateSpace Independent Publishing Platform; 2015.

[23] Jones Eric, Oliphant Travis, Peterson Pearu, et al. SciPy: Open source scientific tools for Python.

[24] Lam Siu Kwan, Pitrou Antoine, Seibert Stanley. Numba: A LLVM-based python JIT compiler. In: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC - LLVM '15, 2015, p. 1–6.

[25] Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng 2007;9(3):90–5.

[26] Orteu Jean-José, Garcia Dorian, Robert Laurent, Bugarin Florian. A speckle texture image generator. In: Speckle06: Speckles, From Grains to Flowers, Vol. 6341. 2006, p. 63410H, September 2006.

[27] Olufsen Sindre Nordmark, Clausen Arild Holm, Breiby Dag Werner, Hopperstad Odd Sture. X-ray computed tomography investigation of dilation of mineral-filled PVC under monotonic loading. Mechanics of Materials 2020;142:103296.

[28] Pérez Fernando, Granger Brian E. IPython: A system for interactive scientific computing python: An open and general- purpose environment. Comput Sci Eng 2007;9(3):21–9.

[29] Perlin Ken. Improving noise. ACM Trans Graph 2002;21(3):681–2.

[30] Piegl Les, Tiller Wayne. The NURBS Book. Monographs in Visual Communication, Springer Series; 1997.

[31] Vend Roux G, Knauss WG. Submicron deformation field measurements: Part 2. Improved digital image correlation. Exp Mech 1998;38(2):86–92.

[32] Sutton Michael A, Schreier Hubert W, Braasch Joachim R. Systematic errors in digital image correlation caused by intensity interpolation. Opt Eng 2000;39(November 2000):2915–21.

[33] Tustison Nicholas J, Amini Amir A. Analysis Of 4-D Cardiac Mr Data With Nurbs Deformable Models: Temporal Fitting Strategy And Nonrigid Registration. New York: Springer; 2007, p. 493–534.