GPU accelerated toolbox for real-time beam-shaping in multimode fibres

M. Plöschner,[1,3,*] B. Straka,[2] K. Dholakia,[3] and T. Čižmár[1,3]

[1] *Division of Physics, University of Dundee, Nethergate, Dundee, DD1 4HN, Scotland, UK*

[2] *Fakulta strojního inženýrství, Vysoké učení technické v Brně, Technická 2896/2, 616 69 Brno, Czech Republic*

[3] *SUPA, School of Physics and Astronomy, University of St Andrews, St Andrews, Fife, KY16 9SS, Scotland, UK*

[*] *mploschner@dundee.ac.uk*

Abstract: We present a GPU accelerated toolbox for shaping the light propagation through multimode fibre using a spatial light modulator (SLM). The light is modulated before being coupled to the proximal end of the fibre in order to achieve arbitrary light patterns at the distal end of the fibre. First, the toolbox optimises the acquisition time of the transformation matrix of the fibre by synchronous operation of CCD and SLM. Second, it uses the acquired transformation matrix retained within the GPU memory to design, in real-time, the desired holographic mask for on-the-fly modulation of the output light field. We demonstrate the functionality of the toolbox by acquiring the transformation matrix at the maximum refresh rate of the SLM - 204 Hz, and using it to display an on-demand oriented cube, at the distal end of the fibre. The user-controlled orientation of the cube and the corresponding holographic mask are obtained in 20 ms intervals. Deleterious interference effects between the neighbouring points are eliminated by incorporating an acousto-optic deflector (AOD) into the system. We remark that the usage of the toolbox is not limited to multimode fibres and can be readily used to acquire transformation matrix and implement beam-shaping in any other linear optical system.

OCIS codes: (060.2350) Fiber optics imaging; (090.1000) Aberration compensation; (090.1760) Computer holography.

References and links

1. A. J. Thompson, C. Paterson, M. A. Neil, C. Dunsby, and P. M. French, "Adaptive phase compensation for ultracompact laser scanning endomicroscopy," Opt. Lett. **36**, 1707–1709 (2011).
2. B. A. Flusberg, J. C. Jung, E. D. Cocker, E. P. Anderson, and M. J. Schnitzer, "In vivo brain imaging using a portable 3.9 gram two-photon fluorescence microendoscope," Opt. Lett. **30**, 2272–2274 (2005).
3. B. A. Flusberg, E. D. Cocker, W. Piyawattanametha, J. C. Jung, E. L. M. Cheung, and M. J. Schnitzer, "Fiber-optic fluorescence imaging," Nat. Methods **2**, 941–950 (2005).
4. R. Di Leonardo and S. Bianchi, "Hologram transmission through multi-mode optical fibers," Opt. Express **19**, 247–254 (2011).
5. T. Čižmár and K. Dholakia, "Shaping the light transmission through a multimode optical fibre: complex transformation analysis and applications in biophotonics," Opt. Express **19**, 18871–18884 (2011).
6. T. Čižmár and K. Dholakia, "Exploiting multimode waveguides for pure fibre-based imaging," Nat. Commun. **3**, 1027 (2012).
7. I. N. Papadopoulos, S. Farahi, C. Moser, and D. Psaltis, "Focusing and scanning light through a multimode optical fiber using digital phase conjugation," Opt. Express **20**, 10583–10590 (2012).

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

8. Y. Choi, C. Yoon, M. Kim, T. D. Yang, C. Fang-Yen, R. R. Dasari, K. J. Lee, and W. Choi, "Scanner-free and wide-field endoscopic imaging by using a single multimode optical fiber," Phys. Rev. Lett. **109**, 203901 (2012).
9. I. N. Papadopoulos, S. Farahi, C. Moser, and D. Psaltis, "Increasing the imaging capabilities of multimode fibers by exploiting theproperties of highly scattering media," Opt. Lett. **38**, 2776–2778 (2013).
10. I. N. Papadopoulos, S. Farahi, C. Moser, and D. Psaltis, "High-resolution, lensless endoscope based on digital scanning through a multimode optical fiber," Biomed. Opt. Express **4**, 260–270 (2013).
11. R. Nasiri, Mahalati, R. Y. Gu, and J. M. Kahn, "Resolution limits for imaging through multi-mode fiber," Opt. Express **21**, 1656–1668 (2013).
12. Y. Choi, C. Yoon, M. Kim, J. Yang, and W. Choi, "Disorder-mediated enhancement of fiber numerical aperture," Opt. Lett. **38**, 2253–2255 (2013).
13. S. Bianchi and R. Di Leonardo, "A multi-mode fiber probe for holographic micromanipulation and microscopy," Lab Chip **12**, 635–639 (2012).
14. S. M. Popoff, G. Lerosey, R. Carminati, M. Fink, A. C. Boccara, and S. Gigan, "Measuring the transmission matrix in optics: An approach to the study and control of light propagation in disordered media," Phys. Rev. Lett. **104**, 100601 (2010).
15. A. M. Caravaca-Aguirre, E. Niv, D. B. Conkey, and R. Piestun, "Real-time resilient focusing through a bending multimode fiber," Opt. Express **21**, 12881–12887 (2013).
16. D. B. Conkey, A. M. Caravaca-Aguirre, and R. Piestun, "High-speed scattering medium characterization with application to focusing light through turbid media," Opt. Express **20**, 1733–1740 (2012).
17. M. Cui, "A high speed wavefront determination method based on spatial frequency modulations for focusing light through random scattering media," Opt. Express **19**, 2989–2995 (2011).
18. D. Preece, R. Bowman, A. Linnenberger, G. Gibson, S. Serati, and M. Padgett, "Increasing trap stiffness with position clamping in holographic optical tweezers," Opt. Express **17**, 22718–22725 (2009).

1. Introduction

The last decade witnessed an outstanding effort in bio-medical photonics to increase penetration depth while minimising collateral damage, whether caused by radiation or mechanical influences [1, 2]. Endoscopes and fibre-based devices have assisted extensively to fulfill these needs [3] and reducing the footprint of fibre based devices is therefore of paramount importance, especially for *in vivo* applications. In this regard, there has been a recent step change in this field with the advance of the shaping of light through thin multimode fibres (MMF) [4–7] whilst retaining the information capacity of bulky single mode fibre bundles. There has been progress towards the miniaturisation of endoscopic [1, 6, 8–12], and micromanipulation [5, 13] tools utilising the MMF. However, the applications of multimode fibre technology are hindered by complex propagation characteristics of light inside the fibre. For example, the diffraction limited spot at the fibre input excites many propagating modes inside the fibre. When the excited modes arrive at the other end of the fibre - their phase, amplitude and polarisation appear randomised and we observe a speckle-like output pattern. Although seemingly random, the output is, in fact, a combined result of mode dispersion and geometry dependent mode coupling that can be fully described by a linear transformation matrix. In essence, the knowledge of the transformation matrix of the fibre allows to couple such a field to the proximal end of the fibre that the desired pattern is generated at the distal end [14].

It was recently shown that the input-to-output mode transformation can be measured for a specific spatial configuration of the fibre [5, 6, 8, 13] and the result used to generate desired patterns at the distal end of the fibre, or vice-versa, for imaging the distal end pattern at the fibre input site. However, previous implementations of the transformation matrix acquisition suffered from synchronisation issues between CCD and SLM which prevented the operation of the hardware at the maximum possible speed. We resolve the synchronisation issues in the toolbox by harnessing the computational power of modern GPUs that are able to deliver the mask on the SLM screen without substantial delay. This allows us to acquire the transformation matrix at the maximum possible refresh rate of our SLM (Boulder Nonlinear Systems - 512x512). The speed gain is one of the important features of the toolbox as the full acquisition of the transformation matrix is a time consuming process that has to be repeated each time

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

the fibre configuration is changed. To the best of our knowledge, our toolbox offers the fastest implementation of transformation matrix acquisition using solely the SLM during the transformation matrix measurements. We note that more rapid acquisition times were achieved using a Digital Micromirror Device [15, 16] or combination of two AODs and SLM [17], however, the methods measured just a fraction of the full transformation matrix, giving capability to display only a single point at the fibre output. In contrast, our toolbox measures the full transformation matrix which allows us to display any pattern at the fibre output.

The generation of output patterns used (e.g. in micromanipulation studies [5, 13]) normally requires a substantial computational time. As a result, a sequence of pre-calculated masks has to be used to realise such manipulation which precludes convenient interactive engagement with the system and is far from a real-time implementation. Superposition of holograms generating an array of output spots also suffers from undesired interference. In our presented toolbox, we address both of these issues. The full transformation matrix of the multimode fibre is stored in the GPU memory and, as a result, the holograms corresponding to the desired patterns can be computed on-demand in approximately 20 ms for 120 output points. Unfortunately, the interference effects of neighbouring spots, especially in closed packed patterns, decrease the quality of the output. This effect can only be partially removed using the iterative Gerchberg-Saxton algorithm [5]. In our toolbox, we achieve a complete removal of the interference in the output patterns by implementing an acousto-optic deflector (AOD) into the setup [6]. This allows to display the fibre output points in a time-discrete manner, preventing interference. The whole AOD cycle, for 120 points, is 0.25 ms long, which is much shorter then the time required for hologram calculation.

This paper is divided in the following way: we first describe the principles of transformation matrix measurement. We then proceed to describe the experimental setup for the generation of arbitrary output patterns with implemented AOD in the system. Finally, we demonstrate the functionality of the GPU toolbox by implementing a routine to generate, in real-time, an interference free, fully interactive 'rotating' cube made of 120 points at the distal end of the MMF fibre. The source code along with the guidelines on how to install the toolbox, use and modify the software are presented in the supplementary information. We believe the toolbox will enable a wider uptake of this emergent technology by MMF community.

2. Principles of transformation matrix measurement

The measurement of the transformation matrix is performed by scanning a diffraction limited spot across the input facet of the fibre. Each spot position, (u,v), is addressed by a specific diffraction grating on the SLM (see Fig. 1). This method differs from our previous implementation, where the SLM was divided into discrete regions, each of which represented a plane-wave incident at some specific angle on a fibre input facet. This new implementation, inspired by [13], has several advantages. Firstly, it utilises the whole area of the SLM during the calibration process, thus using the maximum laser power for each spot. Second, it provides a direct means of mode discretisation of the input facet of the fibre by assigning a specific grating to each mode on the orthogonal grid (u,v). Finally, the optimised holograms are at the full resolution of the SLM by default, without the need to create very small discrete regions on the SLM.

Each spot position (u,v), at the input facet, excites a particular linear superposition of propagating modes inside the MMF. The propagating modes subsequently create certain field pattern at the distal end of the fibre having coordinates (x,y). The resulting pattern is then interfered with the collimated Gaussian output of a single-mode fibre (SMF) [5]. The simultaneous coupling of light into SMF and MMF is realised on the SLM by a complex superposition of corresponding gratings and taking only the phase term of the resulting complex matrix. The phase of the spot in (u,v) is subsequently altered N times over a range of $2\pi n$. The images (120×120

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

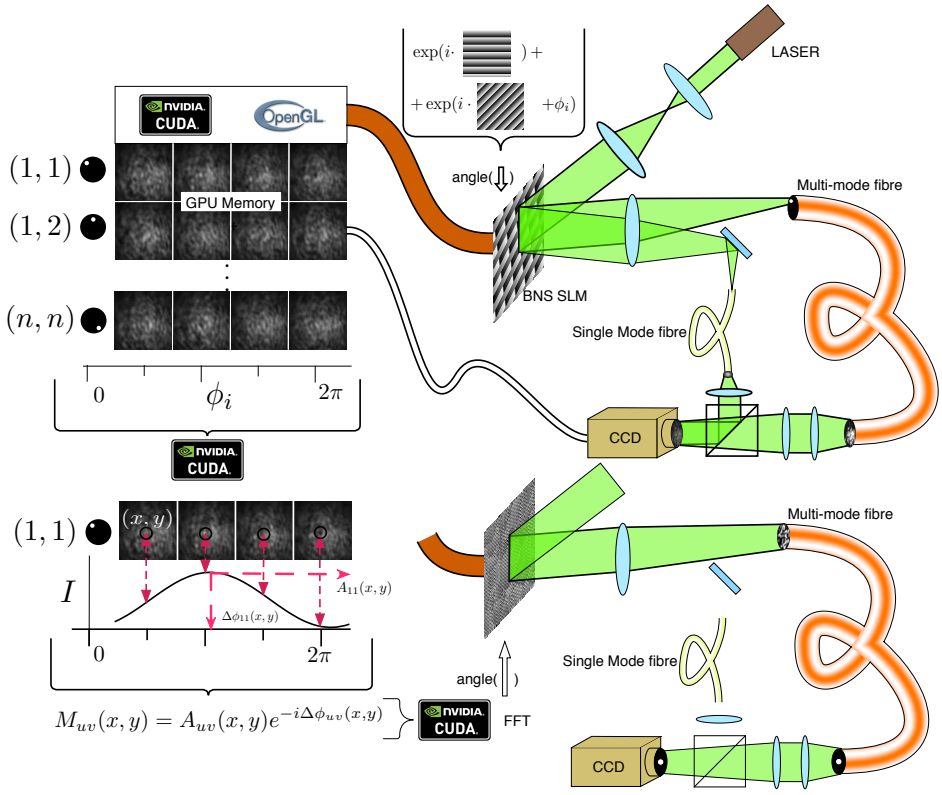

Fig. 1. Laser light from Verdi (5 W; $\lambda = 532$ nm) is magnified by a telescope in order to overfill the SLM plane. The complex superposition of two gratings and subsequent extraction of phase from the resulting complex matrix is applied on the phase-only SLM. The first grating generates a spot in (u,v) at the input facet of MMF (Thorlabs M14L01), the second grating focuses a spot on the input to SMF. The output fields from SMF and MMF are interfered on the CCD and the image is sent to the GPU memory. The phase of the spot in (u,v) is changed N times over a $2\pi n$ range and the whole process is repeated for all input modes (u,v) of the MMF. The resulting images are used to extract the phase and amplitude matrix $M_{uv}(x,y)$ of the output field in any point (x,y). Applying FFT to $M_{uv}(x,y)$ creates a hologram generating field at the fibre input side that results in a well defined spot in (x,y) at the fibre output.

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

pixels) acquired by the CCD (Basler piA640-210gm) for each of these phase steps are subsequently saved in the GPU memory. The resulting images can be used to extract the input (u,v)-mode phase $\Delta\phi_{uv}(x,y)$ and amplitude $A_{uv}(x,y)$ information for the output field in any point (x,y). The process is repeated for all input spot coordinates (u,v). The well defined spot in arbitrary (x,y) can be generated by constructive interference of the output fields in (x,y). To do that we have to adjust the phase and amplitude of the input modes by multiplying the gratings generating them by a complex number $A_{uv}(x,y)e^{-i\Delta\phi_{uv}(x,y)}$ and sum all such adjusted gratings in a complex manner. This process is intuitive but relatively slow. An equivalent approach is to FFT the complex matrix $A_{uv}(x,y)e^{-i\Delta\phi_{uv}(x,y)}$ which leads directly to the desired SLM hologram. The FFT process is fast enough to calculate a hologram focusing the spot into an arbitrary position (x,y) in real-time from the raw acquired data. However, it is much faster to save the input modes amplitude and phase matrices $M_{uv}(x,y) = A_{uv}(x,y)e^{-i\Delta\phi_{uv}(x,y)}$ for all output points (x,y) and calculate the holograms by applying FFT to the $M_{uv}(x,y)$ matrix itself. The FFT step is unavoidable for current generation GPUs as the complete set of holograms for all output points exceeds the amount of memory for most top of the range cards.

The measurement of the transformation matrix for 25×25 input modes for $N = 8$ and $n = 2$, and both input polarisations, takes approximately 49 s when running at a maximum refresh rate of the SLM (204 Hz). In the toolbox, we do not measure the transformation matrix for both output polarisations but this can be easily implemented if required for a specific application. The above corresponds to 10000 frames that require the acquisition of the output fields to be carefully synchronised with the CCD. A routine is implemented into our software to check if the synchronisation of the CCD and the SLM is present. This routine is based on a rotating spot that allows for unambiguous matching of the CCD frames to the SLM holograms. This routine revealed a constant delay of two frames between the displaying of the hologram and the acquisition of the hologram-related frame in our system. This delay can probably be attributed to several software and hardware buffers in our system. However, as the delay is constant, its effect can easily be addressed within the software.

Another problem addressed by the toolbox during acquisition is the phase change of external SMF reference. Even though the acquisition time of the complete transformation matrix is relatively short, the change in phase of external SMF reference occurs due to system drift, small temperature changes and also due to small changes in peak laser wavelength. This phase shift can be significant and therefore cannot be ignored. For this reason the phase of the SMF reference has to be monitored. The phase monitoring of SMF consists of three steps. First, we calibrate the system by recording a series of frames in which the SMF signal interferes with the central input spot (u_c,v_c). The phase of the central spot is modulated over a range of $\langle 0,2\pi\rangle$ in this step. The phase of the SMF external reference is basically constant in this calibration step as the acquisition of frames takes only few milliseconds. In the second step, we always display the input mode at (u,v), cycle the phase through $\langle 0,2\pi n\rangle$ and record the frames. After each individual mode measurement, we also display the central mode (u_c,v_c) with $\phi = 0$ and record the resulting reference frame. In the last step, after finishing the recording of all modes, we utilise the difference between the reference and calibration frames along with the least square method to deduce the change of external reference phase that occurred during the individual mode measurements. The periodic re-displaying of the reference central spot at (u_c,v_c) therefore gives us the necessary knowledge of external reference phase evolution.

3. AOD implementation

The holograms generating focused spots at arbitrary (x,y) points at the fibre output can be superposed in a complex way to generate arrays of spots at the fibre output. However, if the output spots are close to each other, they start to interfere with each other. These undesired

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

interference effects can be partially removed by employing the G-S algorithm to the hologram [5]. As with any other linear optical system the G-S algorithm removes some of the undesired interference effects, but it cannot remove the interference effects completely leaving speckled character of the resulting image. In other words, as long as the output fields are generated as coherent superposition of output modes, interference effects will always negatively influence the output image. This undesirable effect can be eliminated by a method discussed in [6] that constructs the output field point by point in time-discrete mode using the acousto-optic deflector (AOD) (see Fig. 2). The hologram representing the desired output image is created by a standard

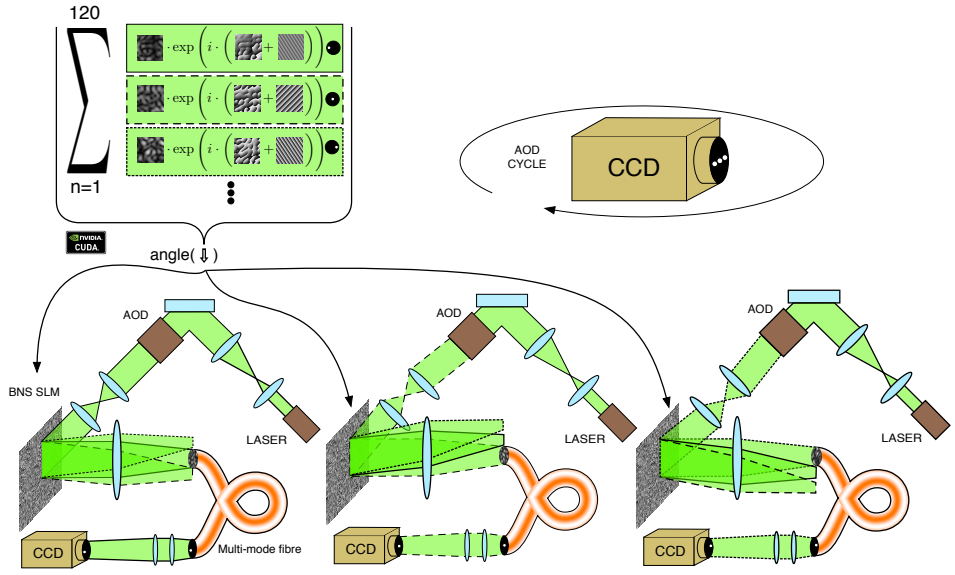

Fig. 2. The beam deflected by AOD impinges on the SLM at an angle given by the current frequency of AOD. The hologram on the SLM consists of the complex superposition of masks focusing light to different spots (x,y) at the fibre output. Each of the constituting masks has an added grating that cancels out one specific angle introduced by the AOD. Subsequently, only the input field corresponding to a mask with the right correction angle for a given AOD deflection will propagate through the fibre. In this way we can cycle between the spots at the output with a maximum refresh rate of the AOD without introducing any interference effects between the output spots.

complex superposition of masks generating the focused spots in (x,y). However, we carefully select the grating of the individual masks such that the effect of the angled incidence of the beam on the SLM, introduced by current AOD frequency, is exactly cancelled for one individual mask from the complete set. As a result, only the mask with correct grating for the current deflection of AOD is imaged on the fibre input facet - all other masks generate fields that miss the input facet for the current AOD deflection. As the AOD performs a full cycle scan through a defined set of frequencies, all the masks contained in the hologram are sequentially addressed, rendering the user defined image at the fibre output.

The complete hologram generation process, for 120 output points, can be done at a refresh rate of 50 Hz enabling interference free images to be generated on-the-fly at the output of the MMF for the first time to our knowledge.

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

4. Real-time pattern generation

To demonstrate the real-time, on-the-fly generation of output images at the end of MMF, we supply a routine, within the GPU toolbox, that displays a rotating cube at the end of MMF (see Fig. 3). This routine is completely interactive allowing the user to change the size of the

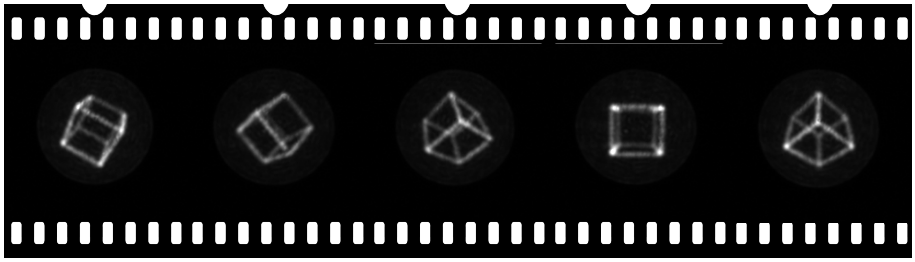

Fig. 3. Projected 3D cube, created using 120 output points, generated at the distal end of the MMF with a refresh rate of 50 Hz. The contrast ratio of the cube lines to the fibre background light is 20 : 1 (Media 1).

cube and its rotational orientation at the refresh rate of 50 Hz. To achieve this refresh rate we FFT the $M_{uv}(x,y)$ matrices for all desired output points (x,y) with 128×128 resolution, scale to native SLM resolution of 512×512 and finally add the gratings. The full-resolution FFT at 512×512 and FFT at 256×256 allows refresh rate of 23 Hz and 42 Hz respectively, with essentially identical contrast as seen in Fig. 4. Even the 64×64 FFT provides rotating cube

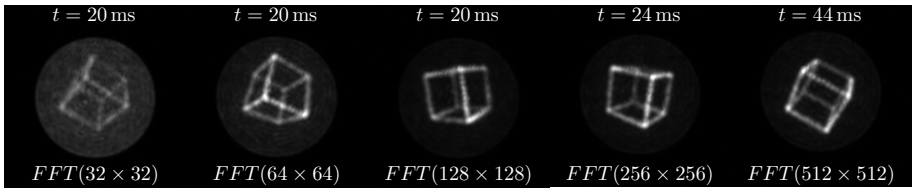

Fig. 4. The FFT resolution influences the contrast and computational time of the generated pattern. The contrast is significantly worse only for 32×32 and 64×64 FFT and the computational time is significantly longer only for 512×512 FFT case. Therefore, the optimal trade-off between speed and contrast is for 128×128 or 256×256 FFT.

with little change in contrast compared to the 512×512 FFT. The only significantly worse case is 32×32 FFT that suffers from a significant loss of contrast possibly due to fibre coupling of neigboring orders of the created grating. All the FFT resolution cases, up to 256×256 have roughly the same computational time of 20 ms. The constant computational time can be explained by function overhead prohibiting lower computational times. The optimum trade-off between speed and quality is therefore at the FFT resolution of either 128×128 or 256×256. The routine can be easily supplied with any user-defined pattern with any number of output points supported by the fibre and AOD system. As the computational complexity appears to increase only after 256×256 FFT, the high refresh rate for larger number of points than presented here, can be maintained by decreasing the resolution of FFT.

5. Conclusion

We have designed an open-source GPU accelerated toolbox for shaping the light transmission through a MMF. The toolbox has several modalities. It allows fast and complete determination

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

of transformation matrix of the MMF for both input polarisations. The results of the transformation matrix are stored in GPU memory and the data can be used to generate arbitrary patterns at the distal end of the fibre in real-time. The interference effects between neighbouring points at the output facet were addressed by implementation of AOD into the system. The AOD allows for an extremely fast, but crucially time-discrete display of the output spots, which completely eliminates the interference effects from the output pattern. The GPU routine driving the system with AOD in real-time, and generating holograms on-the-fly, was demonstrated on a fully interactive model of rotating cube being displayed at the distal end of the MMF. The routines in the GPU toolbox are not limited to MMFs. The transformation matrix measurement, and the output image formation can be utilised in any linear optical geometry, whether it is an ideal Fourier system, system suffering from aberrations or an entirely random system as the multimode fibre used in this study. As such our GPU toolbox should prove useful for a wide community of researchers interested in exploiting complex media and particularly for those wishing to exploit the MMF in endoscopic imaging applications.

6. Appendix - Supplementary information for GPU toolbox

6.1. General information

The holograms are computed on the GPU using **Computing Programming Language** called CUDA C. This language is very similar to standard C as it gives very direct representation of the underlying hardware (e.g. memory pointers, shared memory). This makes the computations on the holograms very straightforward compared to the OpenGL based calculations of the holograms [18](Red Tweezers software from Miles Padgett Group - http://www.gla.ac.uk/schools/physics/research/groups/optics/research/opticaltweezers/). The results of the CUDA C can be easily displayed in a **Shading Programming Language**, like OpenGL. This is done by feeding the OpenGL pixel buffer with a memory pointer pointing to a hologram generated in CUDA C. The CUDA C and OpenGL code running on the GPU are wrapped into a standard C code and all the GPU-based functionalities are exported into a single Dynamic Linking Library (DLL) called `GPU_hologram.dll`. The functions in the library can be used by any external program. In our case, we have created a high level User Interface in LabView calling the functions in `GPU_hologram.dll`.

6.2. Hardware requirements

6.2.1. Graphics cards supported

The code runs on devices with Compute Capability 2.x and higher. We provide several versions of the code for download at http://complexphotonics.dundee.ac.uk/. The supplied code is compiled using the Compute Capability 2.0/3.5 that is available on the latest NVIDIA cards (GTX 570/GTX 780). However, the code can be easily recompiled for any Compute Capability starting from 2.x. Please note that the GPU performing CUDA C calculations and OpenGL output should not be used for standard monitor output. Separate card for monitor output is recommended.

6.2.2. CCD and SLM setup

The system was tested with Basler CCD (piA640-210gm) with the latest firmware version and a Boulder Nonlinear Systems (BNS 512×512) SLM. Please use the camera settings in the supplied icd file in LabView MAX to guarantee synchronisation with SLM. In principle, any CCD and SLM can be synchronised by using the delay test routine supplied with the toolbox.

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

6.3. Installation guide

The following installation guide was tested on Windows 7 Professional 64bit. Please follow the installation guide very carefully in order to make the system fully functional.

6.3.1. Setting up the development environment

In order to modify the supplied code we recommend installing Microsoft Visual Studio 2010 Professional Edition or Professional version supports the NVIDIA NSight 3.1 developer platform that automatically setups the Visual Studio environment for CUDA C and provides useful debugging tools. The Professional version also supports the compilation of 64bit libraries. The provided code can be compiled in any Microsoft Visual Express Edition supported by the CUDA Toolkit. The Express Edition requires manual setup of the development environment for CUDA C that is not described in this paper. Furthermore, the Express Edition does not support the compilation of 64bit libraries which means that 32bit version of LabView has to be used.

It is important to install Service Pack 1 (VS10sp1-KB983509) for Visual Studio 2010 otherwise the code will not compile.

The NVIDIA NSight 3.1 is supposed to setup syntax highlighting in Visual Studio. However, this was not the case in our system. If you encounter the same problem, go to

```
C:\ProgramData\NVIDIA Corporation\CUDA Samples\
v5.5\doc\syntax_highlighting\visual_studio_8\
```

and copy the file *usertype.dat* into

```
C:\Program Files (x86)\Microsoft Visual Studio 10.0\Common7\IDE
```

Lastly, the IntelliSense feature of Visual Studio 2010 does not work very well with CUDA C as it is primarily destined to work with C and C++. As a result kernel calls using <<< and >>> are giving errors. Furthermore some other errors and warnings are identified wrongly. To turn of IntelliSense go to *Tools* → *Options* → *Text Editor* → *C/C++* → *Advanced* → *IntelliSense* and set the feature *Disable IntelliSense* to True.

6.3.2. GPU setup

- Install the latest NVIDIA GPU driver. (system tested for 331.65)
- Open the NVIDIA Control panel and go to *3D Settings* → *Manage 3D Settings* → *Global Settings* and change **CUDA - GPUs** to your main computation card. Also turn on Vertical Synchronisation. This is necessary to avoid tearing in the generated holograms. You can also programatically select the CUDA C card based on the higher Compute Capability directly in our LabView interface but this might fail in case two identical cards are present in the system.
- Install the latest CUDA Toolkit available (system tested for CUDA Toolkit 5.5). This will also install Nsight Visual Studio Edition (version 3.1), which automatically setups the development environment in Microsoft Visual Studio 2010.
- To verify the installation go to (`ProgramData` is a hidden folder)

  ```
  C:\ProgramData\NVIDIA Corporation\CUDA Samples\v5.5\
  Bin\win64\Release
  ```

  and run the program *bandwidthTest.exe*. The program should return PASS. Also run the program *simpleGL.exe*. This verifies the OpenGL CUDA interoperability that we rely on.

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

6.3.3. LabView setup

The Labview 2013 Full or Professional Development System (32bit/64bit) is required to run our GPU code. Only these versions of LabView contain licenses to use the NI LabView 2013 GPU Analysis Toolkit (32bit/64bit) which is necessary to interface with DLLs created. You also need to install the NI Vision Acquisition and Development Module to interface with the camera. Due to the extreme synchronisation requirements of the system, we also highly recommend to run LabView in highest priority mode to avoid any interference from lower priority routines running on the system. Finally, you also need to install the NI GPU Analysis Toolkit 2013 (32bit/64bit).

6.3.4. OpenGL setup

Download *freeglut-MSVC-2.8.1-1.mp*, *glew-1.10.0-win32* and *glut-3.7.6-bin-32and64*. The files contain both 32bit and 64bit files.

- Copy all 64bit DLL files into `C:\Windows\System32\`. This might look confusing as one would expect 32bit DLLs to reside in System32 folder, but 64bit DLLs are indeed located in System32 folder.
- If you work on a 32bit system copy the 32bit DLLs into `C:\Windows\SysWOW64\`
- Copy all the 64bit `.lib` files (this might include files with 32 in their name) into

  ```
  C:\Program Files (x86)\Microsoft Visual Studio 10.0\
  VC\lib\amd64\
  ```

- Copy all the 32bit `.lib` files into

  ```
  C:\Program Files (x86)\Microsoft Visual Studio 10.0\
  VC\lib\
  ```

- Copy all the header files into

  ```
  C:\Program Files (x86)\Microsoft Visual Studio 10.0\
  VC\include\GL\
  ```

At this point we need to inform the linker about the files we have just added. This has already been done in the project but in case you are setting up a new one right click on the project in the Solution Explorer and go to *Properties* → *Configuration Properties* →*Linker* → *Input* → *Additional Dependencies* and add

```
glew32.lib glu32.lib openGL32.lib freeglut.lib
glut64.lib glut32.lib
```

Finally, go to *Properties* → *Configuration Properties* →*C/C++* → *General* → *Additional Include Directories* and add the path where the GL headers are included

```
C:\Program Files (x86)\Microsoft Visual Studio 10.0\
VC\include\GL\
```

6.3.5. Enabling FFT functionality

In order to be able to use `cufft.h` we need to inform the linker again. Add `cufft.lib` the same way as in the previous point.

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

6.3.6. Setting up Runtime Libraries

By default CUDA C and C compilers use different Runtime Library which causes a linking error. It is necessary to use the same Runtime Library. Right click on the project in the Solution Explorer and go to *Properties* → *Configuration Properties* →*C/C++* → *Code Generation* → *Runtime Library* and change it to *Multi-threaded Debug DLL (/MDd)*. For the release configuration you can choose Multi-threaded DLL (/MD). Now go to *Properties* → *Configuration Properties* →*CUDA C/C++* → *Host* → *Runtime Library* and select `<inherit from host>`. This will ensure that the CUDA C compiler uses the same Runtime Library as the host compiler.

6.3.7. Changing the Compute Capability code generation

You can change the version of Computate Capability into which the code is compiled by right-clicking the project in Solution explorer and going into *Properties* → *Configuration Properties* →*CUDA C/C++*→*Device* and change Code Generation to `compute_xx,sm_xx`. Please be aware though that our code requires at least Compute Capability 2.*x*.

6.4. Calling GPU functions in LabView

The main problem of calling GPU functions from within LabView is the fact that LabView is inherently a multithreaded application whereas our GPU code wrapped in C is single threaded. Multiple calls of C code from within LabView could end up in different threads and there is no simple way for CUDA Contexts running in these threads to share data between each other. **CUDA resources allocated by host (CPU) thread can be accessed only by CUDA calls from the same CPU thread**. For these reasons the calling convention of CUDA C code in LabView is as follows:

1. Create CUDA Context by using `Initialize Device VI`. This VI creates CUDA Context and associates it with the calling thread.
2. Lock the CUDA Context using `Prepare Data For Lock VI`. This ensures that the reference to CUDA Context cannot be accessed, modified or changed by another thread during DLL call.
3. Call the GPU function using `Call Library Function Node` from within the `In Place Element Structure`. The `In Place Element Structure` will ensure that the CUDA Context is accessed in the same data space. Right-click on the `Call Library Function Node` and go to *Configure*→ *Function*→*Thread* and set *Run in any thread*. This seems contradictory as we want the GPU function to be executed in the thread from which the CUDA Context was created. However, the CUDA Context wired to `In Place Element Structure` ensures that the DLL call is processed within the thread of CUDA Context.
4. Use `Prepare Data For Unlock VI` to unlock the CUDA Context reference. CUDA Context can be pushed to other host thread if needed and a failure to unlock would prevent this.

Using the above steps ensures that all the DLL calls are executed within the same CPU thread and can communicate and share data between each other.

7. Using the LabView interface

Open the LabView `CUDA_project.lvproj` file and then open the `cuda_display_masks.vi`. This file is ideal for testing that the whole installation process

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

was successful. In the main interface of the VI select the path to `GPU_hologram.dll` and to `masks.dat` on your system. This is normally in the project folder. The camera features are disabled in this VI to allow the testing of CUDA OpenGL interoperability alone but you can easily re-enable them if you wish. Select the position of the OpenGL window somewhere on the screen and run the VI. If the installation was successful you should be able to see a 512×512 window with a test mask in it. You should also be able to change the masks by moving the slider n that addresses n-th mask in the `masks.dat` file. I also recommend studying this simple program in case you want to build your own modified project as it is quite clear how the device/OpenGL window is initialised, how the GPU memory is allocated and data copied to the GPU.

If you want to learn more about changing the holograms on-the-fly using the LabView interface then open the VI called `cuda_display_masks_float.vi`. This VI is similar to the previous one but it has some added features that allow mask modification. The included features allow to select annular part of the hologram and add quadratic modulation. In case all the above tests were successful you should be ready to open the `calibration.vi` itself.

First, you need to specify the correct path to `GPU_hologram.dll` and also to `gratings.dat`, `AOD_spot_intensity.dat` and `AOD_spot2_intensity.dat` (see Fig. 5). The last three files are necessary for calibration of AOD if present in the system. The `gratings.dat` contains pairs (g_x, g_y) that create gratings cancelling the AOD induced angle in such a way that the beam always enters the fibre. In case you are not using AOD in the system, change all the pairs to your central grating and feed this into the program. The files `AOD_spot_intensity.dat` and `AOD_spot2_intensity.dat` compensate for the intensity fluctuation introduced by the AOD's varying efficiency at different deflections. Again, if you do not have AOD in your system, just change the values in those files to 1. You can also change the FFT resolution, the number of displayed points and SLM resolution in this panel.

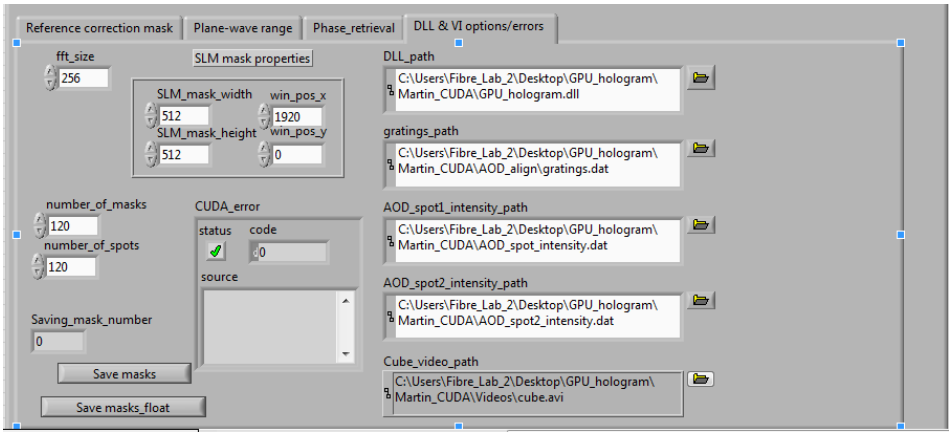

Fig. 5. Options panel allows to setup paths to the DLL library that contains functions to be called and also path to files necessary for AOD operation.

At this stage you should be ready to start the calibration of your system.

7.1. Pre-calibration steps

Prior to any calibration, measure the gamma curve of your SLM and select the 'shift' and 'amplitude' in the VI to correspond to the linear region of your system. Also, run the 'Frame Delay Test' routine to determine the frame delay on your system and to check whether the

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

frame delay remains constant. In case it is constant you should be able to see a rotating point changing position with each frame. This is an crucial step that requires a special camera setup. We provide the relevant CCD icd file in the project folder. Also, the camera should be running at the $30\,\mu$s exposure time in all calibration procedures to avoid loss of synchronisation for acquisitions involving more than 10000 frames.

7.2. Calibration procedure

The calibration consists of several steps outlined in Fig. 6. First, you need to run wavefront correction for the SMF fibre that serves as an external reference. Block the MMF entrance, select the 'Singlemode fibre correction' button and press 'Testing 1,2,3,4'. We recommend using the selected grating for reference of $C_{r_{cpu}} = (0, -0.0625)$ but if this is not possible on yours system change it. Make sure the reference is very bright. We use the beamlet method in this procedure which means only a fraction of intensity will be on the camera once the calibration starts. When this is done, stop the testing, crop the camera ROI by using the 'Default ROI for calibration' and start testing again. If the beam is not centred in the cropped region, change the default ROI or move the beam. Also make sure you use 120×120 pixels in cropped image. Larger images can cause synchronisation issues. We also recommend to set the CCD 'Shutter' time to $30\,\mu$s during all calibration steps to avoid synchronisation issues. If you are satisfied, stop the testing and click 'Do task 1,2,3,4'. This will start the calibration. Once this is finished the corrected amplitude and phase mask are shown. If the images look random and noisy it is highly probable that the light intensity was too low during measurement. Repeat until you get a mask with a nice features. Test the resulting mask by pressing 'Testing' and turning 'Use ref corr' on/off. The ON state should generally give much more light at the output.

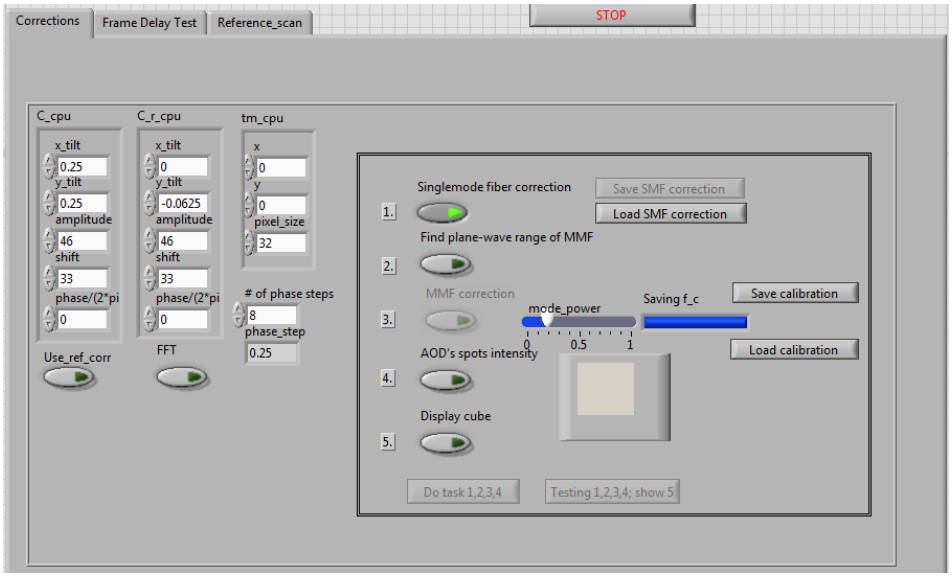

Fig. 6. Main calibration panel

Now you need to make sure that the plane waves you generate on the SLM fully cover the MMF fibre and that the centre plane wave corresponds to the central spot on the fibre. Do this by pressing 'Find plane-wave range of MMF' and pressing 'Do task'. You can also change the range of plane waves and the step in gratings. This in turn changes the number of modes entering the fibre. We have tested the system for 625 modes and we recommend using this value

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

in case you are using the same fibre (Thorlabs M14L01) . The output should give a centrally located circle with a relatively sharp intensity drop. This drop marks the modes that do not couple into the fibre.

Proceed to step 3 - 'MMF correction'. Press the 'Testing' button. You should see the interference of SMF and MMF signal. Changing the position of the slider 'mode power' from $0-1$ allows control over the ratio of powers between SMF and MMF signal. The value of 1 gives only a MMF mode whereas the value of 0 gives only the SMF signal. Move the slider such

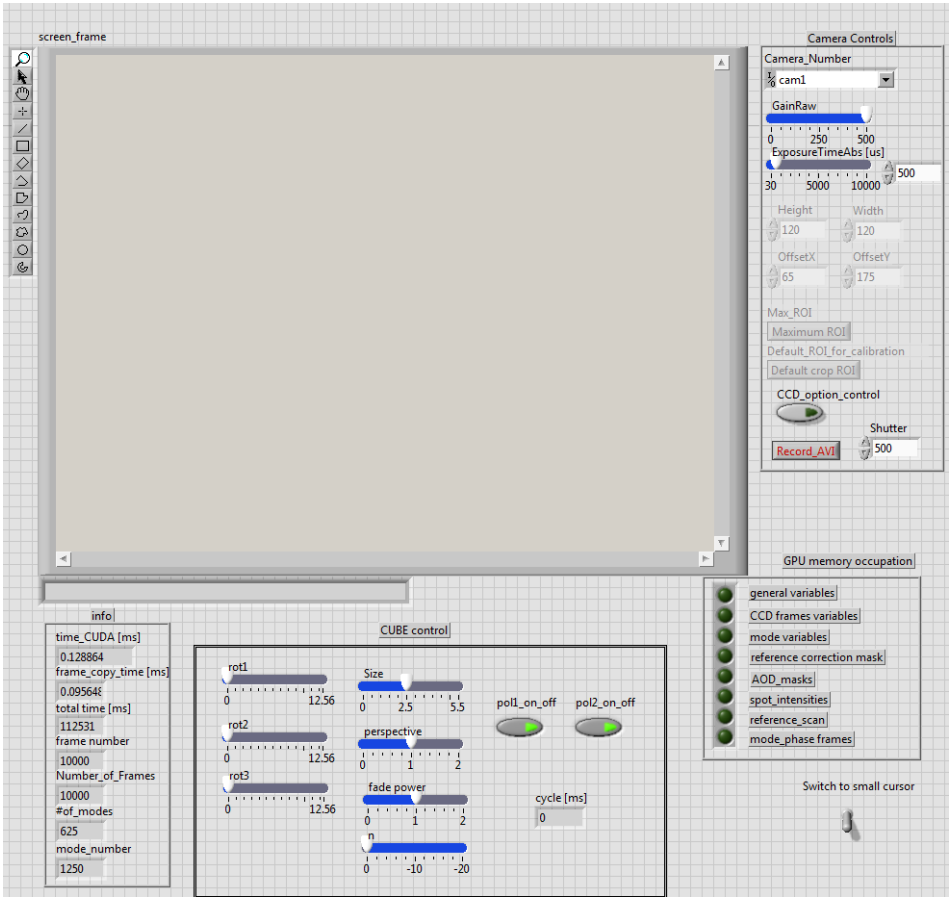

Fig. 7. Cube control and information panel.

that the observed interference pattern just about contains drops in intensity to zero. Be careful not to increase the MMF mode power too much as this would effectively render the use of external reference obsolete. If you are satisfied with the results, stop the testing and press the 'Do task 1,2,3,4' button. This will start the calibration procedure for both input polarisations. Once finished, you can check the quality of calibration by 'Testing' and pressing both 'Use ref corr' and 'FFT' buttons. Moving mouse in the camera output window should give a moving point updating position with the position of your mouse. The point should be nicely focused everywhere except at the edges of the fibre. To stop the test, uncheck both 'Use ref corr' and 'FFT' buttons and stop 'Testing'. In case there is no AOD in the system, you can immediately proceed to point 5. 'Display cube' and use the controls in Fig. 7 to move it around. Please be aware that this will contain strong interference artefacts.

#202243 - $15.00 USD Received 29 Nov 2013; accepted 19 Dec 2013; published 31 Jan 2014

7.3. Additional AOD calibration steps

Display the AOD grid that is used for MMF fibre entrance at some auxiliary camera. Open the `point_correction_vi` in the 'AOD align' folder located in the project folder. This VI is intended to find a wavefront correction for the central point of the AOD grid (aberration free point is useful in the next step of our procedure). Adjust the grid values and other sliders to correspond with your system, select the central point of the grid and press 'Go'. Once the amplitude and phase mask are measured exit the VI and open the `find_gratings_for_aligned_points.vi` In 'x' and 'y' fields enter the grating that moves the upper right point to the centre of the screen and also adjust the exposure time to the full cycle of your AOD. Start the VI. The VI finds such a set of gratings that all the AOD points hit the same pixel on the CCD camera. Exit the VI.

In the `calibration.vi` select point 4. 'AOD's spot intensity' and press 'Do task'. This step will display a field at the entrance to the fibre such that the output is focused in the central spot. The field is sequentially multiplied by all gratings measured in the previous step. Due to AOD's varying efficiency for different deflections, the central output point is not constant but exhibits variation in intensity. The VI measures this variation and subsequently adds the mask for cube hologram in such a way to account for this. Make sure the central output point is not saturated for any grating as this will bias the measurement and lead to point intensity fluctuations in the displayed cube.

Acknowledgments

TC and MP acknowledge support from the University of St Andrews, the University of Dundee and the Scottish Universities Physics Alliance (SUPA). We also thank the UK Engineering and Physical Sciences Research Council for funding. KD is a Royal Society-Wolfson Merit Award Holder. BS contributed to this work within Erasmus Student Mobility for Placement programme funded by European Commission.