Can surgical simulation be used to train detection and classification of neural networks?

Odysseas Zisimopoulos[1] ✉, *Evangello Flouty*[1], *Mark Stacey*[1], *Sam Muscroft*[1], *Petros Giataganas*[1], *Jean Nehme*[1], *Andre Chow*[1], *Danail Stoyanov*[1,2]

[1] *Touch Surgery, Kinosis, Ltd, 230 City Road, EC1 V 2QY, London, UK*
[2] *Computer Science, University College London, Gower Street, WC1E 6BT, London, UK*
✉ *E-mail: odysseas.zisimopoulos@touchsurgery.com*

Published in Healthcare Technology Letters; Received on 25th July 2017; Accepted on 31st July 2017

Computer-assisted interventions (CAI) aim to increase the effectiveness, precision and repeatability of procedures to improve surgical outcomes. The presence and motion of surgical tools is a key information input for CAI surgical phase recognition algorithms. Vision-based tool detection and recognition approaches are an attractive solution and can be designed to take advantage of the powerful deep learning paradigm that is rapidly advancing image recognition and classification. The challenge for such algorithms is the availability and quality of labelled data used for training. In this Letter, surgical simulation is used to train tool detection and segmentation based on deep convolutional neural networks and generative adversarial networks. The authors experiment with two network architectures for image segmentation in tool classes commonly encountered during cataract surgery. A commercially-available simulator is used to create a simulated cataract dataset for training models prior to performing transfer learning on real surgical data. To the best of authors' knowledge, this is the first attempt to train deep learning models for surgical instrument detection on simulated data while demonstrating promising results to generalise on real data. Results indicate that simulated data does have some potential for training advanced classification methods for CAI systems.

1. Introduction: Computer-assisted surgical (CAS) systems augment the surgeon's capabilities through information fusion and presentation during a procedure with the potential for increasing surgical precision, optimising ergonomics and surgical actions, and enhancing patient safety [1]. In CAS, surgical workflow understanding and procedural segmentation into operational phases is a potential step towards reducing surgical errors and improving patient outcomes through the standardisation of processes and techniques much like surgical checklists [2]. The development of cognitive CAS systems that automatically analyse surgical workflow would assist surgeons and medical practitioners, offering solutions to essential tasks in the operating room like phase recognition, surgical assessment, operation monitoring and optimised scheduling [1, 3]. Furthermore, surgical workflow analysis could be beneficial for automated surgical video indexing and operational key step extraction for creating an educational content, a task that is currently manual and time consuming, for disseminating surgical technique or management of specific adverse events.

Automated phase recognition has been studied for a number of years, predominantly through using information about the motion of the surgeon's hands or the instruments in the case of minimally invasive surgery [4, 5]. Such studies indicate that information about the presence of instruments at different procedural time points is valuable for phase recognition. This is reasonable given that different instruments are used to achieve specific procedural tasks. Other studies suggest using surgical triplets (information of the utilised tool, the anatomical structure, and the surgical action) [6] or visual features [7–9] but these features were either manually annotated or hand-crafted, which is not robust and does not generalise well across different surgical indications. More recently with the emergence of deep learning techniques interesting studies are emerging that bypass the need for manual feature tailoring or for explicit instrument segmentation [10]. Though exciting because manual design is avoided, these methods require large amounts of data for training, their performance can be improved with additional developments and it is likely that explicit information about instrument placement and kinematics will be assistive.

Extracting information about the presence and motion of instruments during surgery can be achieved through different sensing modalities. When robotic systems are using encoders, the kinematic chain automatically provides this data and it can be used directly for analysis [11]. In the majority of surgical procedures, however, these instruments do not have motion sensors and in this case when a surgical camera is used (e.g. a laparoscope or a microscope) vision-based approaches are an attractive solution [12]. For vision-based techniques, deep convolutional neural networks (CNN) have established themselves as the new state-of-the-art approach for computer vision problems in image classification [13] and semantic segmentation [14]. A drawback of deep CNNs is the requirement of large training datasets when training from the scratch, however, this can be somewhat mitigated through transfer learning either by using a CNN pre-trained on large datasets, like ImageNet [15], as an off-the-shelf feature extractor, or by fine-tuning a pre-trained network on smaller task-specific datasets [16]. This concept has recently been adopted for tool presence detection and joint surgical phase recognition [17], where a CNN is used as a feature extractor [13], appended by a fully connected classification layer for the task of tool presence detection. Yet the transfer learning aspect of the adaptation of features to the surgical context is still a significant and not fully understood challenge. Additionally, for fully robust systems, the training dataset must encapsulate a very wide range of variable conditions that may occur in practice.

In this Letter, we focus on exploring whether surgical simulation can be used to generate data that is useful for training deep CNN by covering a sufficient domain to facilitate models to be applied to a real surgical video. The contribution of this Letter is to compare the application of two different models for automated tool segmentation in a video both trained on synthetic data and tested on real data as shown in Fig. 1. The first model is the fully convolutional network adaptation (FCN-VGG) [14] trained to perform supervised semantic segmentation in 14 classes that represent the 13 different tools present in our datasets and an extra class for the background of the environment. The second model is the pix2pix for unsupervised domain adaptation which we apply attempting to translate simulated

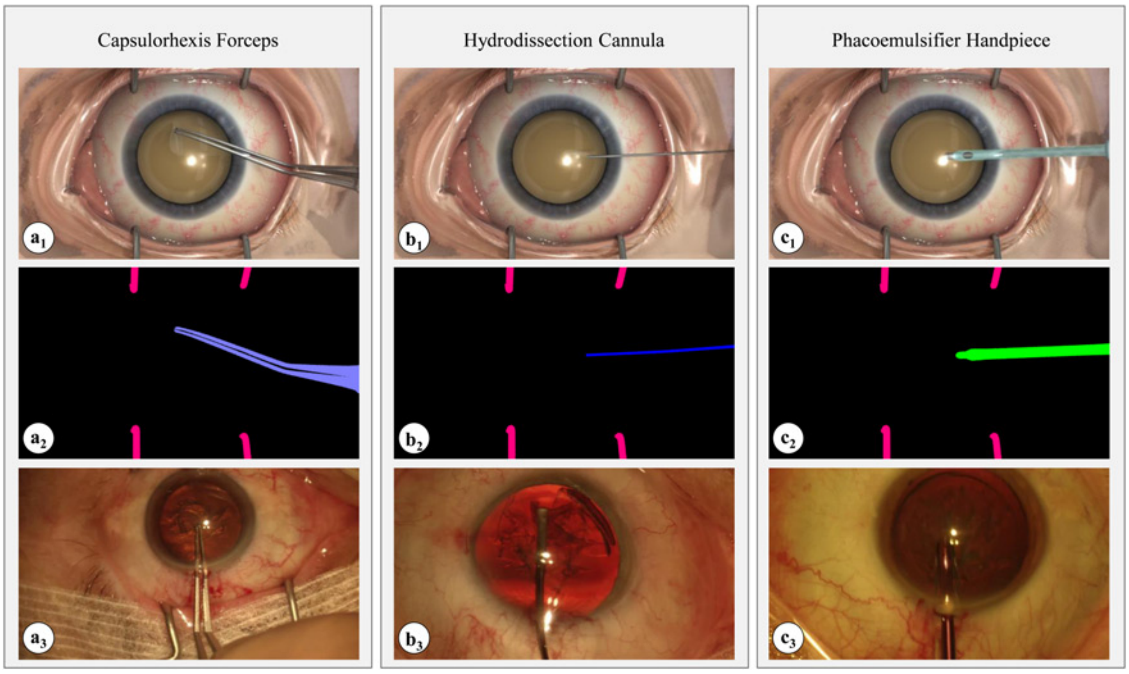

Fig. 1 *Example synthetic (top row) and real (bottom row) datasets used in this work; (a–c) three exemplar tools used in cataract surgery*
a_1, b_1, c_1 Images generated through a commercial simulation environment
a_2, b_2, c_2 Segmentation masks delineating instruments from the operating site
a_3, b_3, c_3 Real cataract surgery images (https://cataracts.grand-challenge.org/)

images directly into semantic instrument segmentations [18]. In both cases, we train the models on a simulated dataset acquired from a commercially available surgical simulator and attempt to adapt the model domain such that it can be used on real cataract images (2017 MICCAI CATARACTS challenge, https://cataracts.grand-challenge.org/). Our goal is to use the simulator's unlimited capability to generate data with variability in camera pose, lighting or instrument motion, to train machine learning models and then directly apply them to detect tools in real cataract videos. Our results show that there is potential for developing this idea, with the pix2pix technique demonstrating that detecting real instruments using models trained on synthetic data is feasible albeit further advances are needed to correctly perform classification into instrument classes.

2. Materials and methods: Using a surgical simulator 'Touch Surgery', we rendered synthetic cataract data using varying rendering parameters (i.e. lighting conditions and viewing angles), as shown in Fig. 1. The simulation environment (including the surgical tools) is credible as it is created and verified by animators and medical officers together [19]. The simulated cataract operation includes three surgical phases: (i) patient preparation, (ii) phacoemulsification, and (iii) insertion of the intraocular lens. For each phase, we chose 15, 10 and 5 different combinations of rendering parameters that resulted in a total of 17,118 rendering views. For each camera pose, we generated a 960 × 540 RGB image along with a tool segmentation depicting each tool with a different colour. These pairs of simulations–segmentations were used to train our machine learning models for tool detection. The generated dataset was divided into 60, 20 and 20% for a training, validation and testing set of 10,376, 3541 and 3201 frames, respectively.

To test the generalisation of our models we used a real cataract dataset gathered from the CATARACTS challenge training dataset. The real dataset consists of 25 training videos of 1920 × 1080 resolution frames annotated with only tool presence information but without the fully segmented instrument. Tools present within the simulated and real datasets slightly differ in number (21 in real and 24 in simulated) and type. For example, Bonn forceps that are found in the real set do not exist in the simulations and, therefore, had to be discarded from training. We gathered a real set with 14 common classes for a total number of 2681 frames. The 13 tool classes co-existing in both datasets are: (i) hydrodissection cannula, (ii) rycroft cannula, (iii) cotton, (iv) capsulorhexis cystotome, (v) capsulorhexis forceps, (vi) irrigation/aspiration handpiece, (vii) phacoemulsifier handpiece, (viii) vitrectomy handpiece, (ix) implant injector, (x) primary incision knife, (xi) secondary incision knife, (xii) micromanipulator and (xiii) vannas scissors. An additional class is used for the background, when no tool is present.

2.1. FCN-VGG: The FCN-VGG architecture extends the original VGG for semantic segmentation by substituting the fully connected output layer of the network with a convolutional layer, which allows faster training while preventing over-fitting [14, 20]. We fine-tuned the single-stream (up-sampling stride of 32) FCN-VGG following the original methodology [14]. The network consists of 16 trainable convolution layers with rectified linear unit (ReLU) activations, some followed by max-pooling layers. The kernels of the convolution and pooling layers are consistently sized at 3 × 3 and 2 × 2, respectively, throughout the whole network.

FCN-VGG is applied like a filter on an input image $\boldsymbol{x}$ and produces an output $\boldsymbol{y}$ of the same dimensions. The kernels of the convolution layers of the network are applied in a moving way across the image preserving the spatial information of the input. The final layer up-samples the output of the network to the input size. The predictions of FCN-VGG are calculated using the softmax function on a pixel level and the loss that is being minimised is the softmax loss

$$\mathcal{L}_{\text{FCN\_VGG}} = -\frac{1}{N}\sum_{n,i,j,c} g_{n,i,j}^{(c)} \log[\phi(w_{n,i,j}^{(c)})], \qquad (1)$$

where N is the batch size, $g_{n,i,j}^{(c)} \in \{0, 1\}$ and $w_{n,i,j}^{(c)}$ are the ground truth and network's prediction, respectively, of class c for pixel

(i, j) in the nth image and $\phi(\cdot)$ is the softmax function

$$\phi(w_{n,i,j}^{(c)}) = \frac{e^{w_{n,i,j}^{(c)}}}{\sum_{c=1}^{C} e^{w_{n,i,j}^{(c)}}}, \tag{2}$$

where C is the number of different classes.

The layers of the network were initialised with weights pre-trained on the PASCAL VOC-11 [21] dataset, except for the last convolution layer which is task-specific and had to be trained from the scratch. This is the layer that performed the 14 class segmentation of cataract instruments. The weights of this layer were initialised from a Gaussian distribution with a mean of 0 and a standard deviation of 0.01 and the biases were initialised to zero. The network was fine-tuned on the simulated dataset using stochastic gradient descent with base learning rates of 10^{-5} and 10^{-10} and the learning rate of the final layer was multiplied by a factor of 10 in order to accelerate the learning process of this layer. For training and testing FCN-VGG, we used an implementation in the Caffe deep learning framework and the publicly available model of FCN-VGG in Caffe Model Zoo [22].

2.2. Pix2pix: An alternative model that performs image domain transfer through the use of a conditional generative adversarial network (cGAN), which performs unsupervised domain adaptation using two networks, one generator and one discriminator, trained in an adversarial way. The generator maps an input noise vector $\boldsymbol{z}$ to an output image $\boldsymbol{y}$: $G{:}\boldsymbol{z} \to \boldsymbol{y}$. In cGANs, the generator conditions on both a noise vector $\boldsymbol{z}$ and an image $\boldsymbol{x}$ and produces an output image $\boldsymbol{y}$: $G{:}\{\boldsymbol{x}, \boldsymbol{z}\} \to \boldsymbol{y}$. The input image comes from the source domain and the output image comes from the target domain's distribution. The pix2pix (P2P) algorithm learns a mapping between the source and the target domain in order to perform image transfer between the two domains [18]. The discriminator in generative adversarial networks (GANs) is commonly a classifier and is trained to infer whether an image comes from the output of the generator (synthetic) or from the target domain (real). Similarly, in cGANs the discriminator is conditioned on an image from the source domain, $\boldsymbol{x}$, and an image $\mathbf{u}$ from either the target domain or the output of the generator $G(\boldsymbol{x}, \boldsymbol{z})$, and classifies it as real or synthetic: $D{:}\{\boldsymbol{x}, \boldsymbol{u}\} \to \{0, 1\}$. The discriminator is trained to recognise synthetic images whereas the generator is trained to render images from the distribution of the target domain and make them unrecognisable for the discriminator to detect synthetic data. In this context, the generator and discriminator are trained adversarially. By the end of the training, the generator is expected to be capable of producing images realistic enough so that the discriminator has a 50% (random) chance of correct classification.

In this Letter, we use the P2P model for image transfer mapping between two domains with the architecture as presented in the original paper [18]. This is a cGAN model with a generator of an U-Net encoder–decoder architecture and skip connections between different layers of the encoder and the decoder. Both the generator and the discriminator consist of a sequence of convolution, batch normalisation and ReLU layer combinations. The loss function that the cGAN is trying to minimise in P2P is

$$\mathcal{L}_{\text{cGAN}} = \mathbb{E}[\log D(\boldsymbol{x}, \boldsymbol{y})] + \mathbb{E}[\log(1 - D(\boldsymbol{x}, G(\boldsymbol{x}, \boldsymbol{z})))], \tag{3}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are images from the source and target domain, respectively, $\boldsymbol{z}$ is a random noise vector, $D(\boldsymbol{x}, \boldsymbol{y}) \in [0, 1]$ is the output of the discriminator and $G(\boldsymbol{x}, \boldsymbol{z})$ is the output of the generator. The generator tries to minimise this equation, whereas the discriminator tries to maximise it. Being a style transfer model, P2P is constrained to produce an output that is close enough to the input in terms of labelling. This is also necessary in order to preserve the annotation of the input to the output image. This constraint is forced through the use of an additional regularizing loss $L1$

$$\mathcal{L}_{L1} = \mathbb{E}[\|\boldsymbol{y} - G(\boldsymbol{x}, \boldsymbol{z})\|_1] \tag{4}$$

so that the overall objective function to be optimised becomes

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cGAN}} + \beta\mathcal{L}_{L1}, \tag{5}$$

where $\beta \geq 0$ is a weight for the trade-off between the cGAN and the regularisation loss.

Contrary to most GANs, in P2P, a patch-based discriminator is used to classify $N \times N$ patches of the image and aggregate a final decision. The discriminator is a fully convolutional network (FCN) and can be applied on arbitrarily sized images, while at the same time training becomes faster and over-fitting is prevented by keeping the number of parameters low. We used a patch size of 70×70, and the discriminator consists of a sequence of four convolution, batch normalisation and ReLU layer combinations and a one-dimensional convolution output to aggregate the decision. This layer is passed into a Sigmoid function that produces a probability of the input being real (from the target domain). In our experiments we consider the domain of the simulated cataract images as the source domain and the domain of the semantic segmentations as the target domain. The cGAN was trained to learn a mapping between a simulated image and a segmentation, thus performing tool detection. After training, the generator was applied to images from the real dataset in order to perform tool detection by transfer learning.

Despite the generator and the discriminator of P2P being FCNs, we chose to use the default 256×256 image size, and therefore, we resized our simulated and real data before training. All layers were trained from the scratch and weights were initialised from a Gaussian distribution with a mean of 0 and a standard deviation of 0.02. Our implementation is based on the open source code, made publicly available by the authors [18], written in Tensorflow.

3. Experimental results: FCN-VGG was trained on the full training set of ~10k images (10,376 images) towards semantic segmentation using stochastic gradient descent with a batch of 16 and a base learning rate of 10^{-10}. We also resized the dataset and trained on 256×256 frames, according to the application of image translation between semantic segmentation and photos of [18]. We called these models FCN-VGG-10K-Large and FCN-VGG-10K-Small, respectively. Finally, we sub-sampled the resized dataset to form a smaller set of 400, 100 and 100 training, validation and testing images, according to the same image translation application of [18]. For this experiment, we trained using a base learning rate of 10^{-5}. We called this model FCN-VGG-400. FCN-VGG-10K-Large and FCN-VGG-10K-Small were trained for around 2000 iterations each, whereas FCN-VGG-400 was trained for 20,000 since we did not use a batch and the convergence was slower.

P2P was trained solely on 256×256 data, on the sub-sampled and the full dataset; we called these models P2P-400 and P2P-10K, respectively. We used the Adam optimiser with the same parameters as [18], *i.e.* batch size of 1, learning rate of 0.0002 and $L1$ loss weight of $\beta = 100$. P2P-400 was trained for 200 epochs that is 80,000 iterations, whereas P2P-10K for 50 epochs that is 500,000 iterations. An overview of our models can be seen in Table 1.

All training and testing were performed on an Nvidia Tesla K80 GPU with 8 GB of memory.

We used the simulated test set to test the task of tool detection on the simulated images. The segmentations predicted by our models are shown in Fig. 2. We can observe that generally the FCN–VGG models classify correctly the retrieved pixels (i.e. assign correct tool labels) creating rougher segmentations,

Table 1 Summary of all deep learning models trained on the simulated datasets with various sizes and resolutions

Model	Resolution	Training set size
FCN-VGG-400	256×256	400
FCN-VGG-10K-Small	256×256	10,376
FCN-VGG-10K-Large	960×540	10,376
P2P-400	256×256	400
P2P-10K	256×256	10,376

whereas P2P misclassifies a few tools but produces finer segmentations for the detected tools. For example, in the fourth row of Fig. 2, both P2P models predict very good segmentations whereas only FCN-VGG-10K-Large out of all FCN-VGG models is close. In the third row, FCN-VGG-10K-Large assigns the correct classes to the retrieved pixels, successfully detecting the tool, but produces a rough outline, whereas P2P-400 creates a finer outline but picks the wrong label (red instead of purple). For the same input, P2P-10K outperforms both FCN-VGG-10K-Large and P2P-400. Overall, FCN-VGG-10K-Large produces the best qualitative results among the FCN-VGG models and P2P-10K is the best style transfer model. Ultimately, it seems that P2P-10K is our best model.

For the quantitative evaluation of the performance of our models on the simulated test set we calculated the metrics used in [14] for semantic segmentation, namely pixel accuracy, mean class accuracy, mean intersection over union (mean IU) and frequency weighted IU (fwIU). The results of the evaluation are shown in Table 2. We notice that the FCN-VGG models achieved better mean accuracy and mean IU, whereas P2P achieved better pixel accuracy and fwIU. Among FCN-VGG and P2P models, FCN-VGG-10K-Large and P2P-10K are highlighted as the best ones, verifying the qualitative results. We notice that P2P-10K achieved a lower mean class accuracy and mean IU than FCN-VGG-10K-Large. This was caused by the fact that whereas P2P detected many tools reliably (e.g. rows 1, 3, 4 and 5 in Fig. 2), there are classes it missed. This can be shown in the second row of Fig. 2, where the majority of the orange tool was detected as the background while the parts of it that were detected as a tool were assigned the wrong class. Hence, the class accuracy

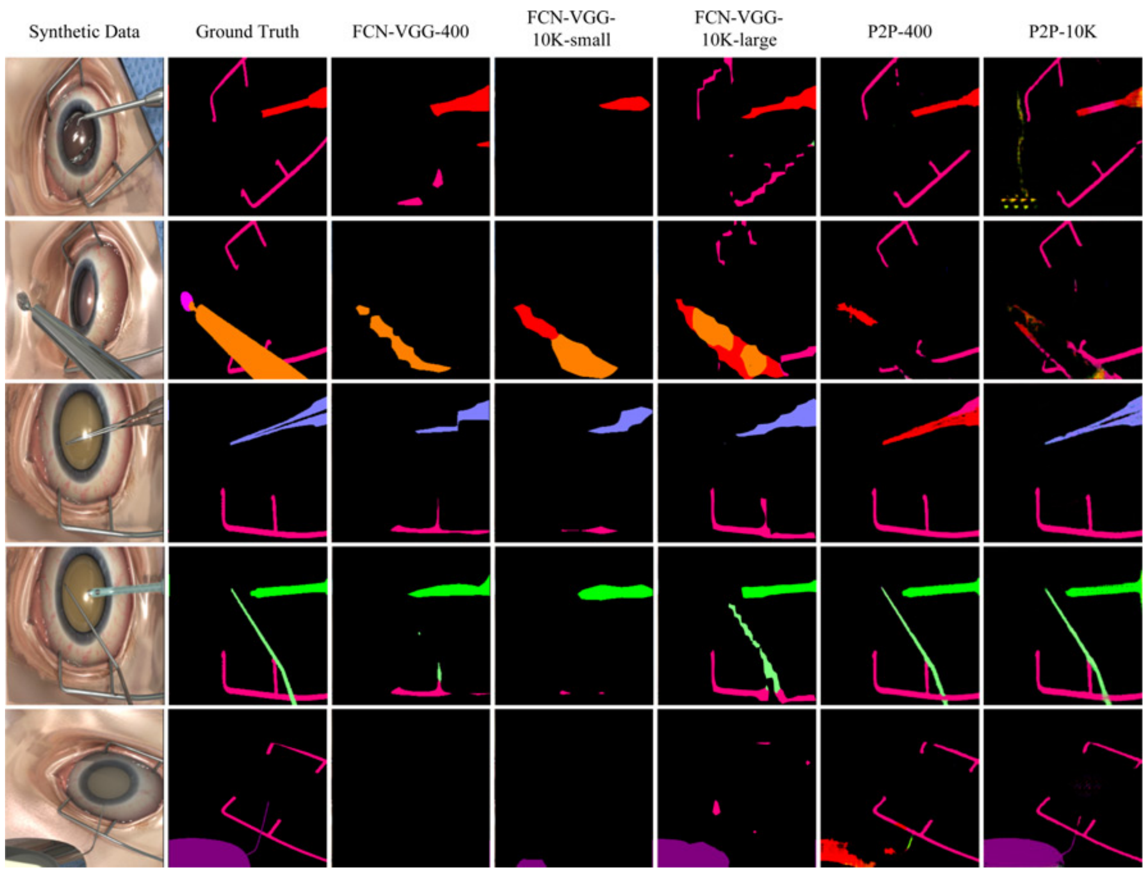

Fig. 2 *Experimental results presenting all the outputs of the two learning models for different parameters (training dataset size and resolution) applied on a simulation testing dataset. The simulated images are shown in the first column and the ground truth segmentation images in the second column*

Table 2 Evaluation metrics for the performance of all deep learning models on the simulated test set

Model	Pixel accuracy	Mean accuracy	Mean IU	fwIU
FCN-VGG-400	0.936	0.334 ± 0.319	0.254 ± 0.297	0.883
FCN-VGG-10K-Small	0.959	0.372 ± 0.355	0.354 ± 0.342	0.922
FCN-VGG-10K-Large	0.977	0.639 ± 0.322	0.526 ± 0.333	0.958
P2P-400	0.981	0.395 ± 0.426	0.196 ± 0.336	0.969
P2P-10K	0.982	0.503 ± 0.363	0.260 ± 0.350	0.974

and IU for this case were close to zero. This was the case for all consecutive frames of the same tool, reducing the mean class accuracy and mean IU. On the other hand, FCN-VGG-10K-Large created rougher segmentations across all tools but had a lower chance of misclassification. This is why P2P-10K has a better fwIU (IU averaged by the real distribution of the classes, ignoring zero IUs) than FCN-VGG-10K-Large.

We have to note that while FCN-VGG performed pixel-level classification by predicting tool labels, P2P performed image translation by generating pixel RGB values. Therefore, we had to apply a threshold to the segmentations of P2P in order to get the final pixel labelling. Although this procedure did not significantly affect the final outcome, it induced some noise in the prediction which could have an effect in decreasing the metrics for P2P.

After training our models on the simulated dataset, we compared their performance for tool detection in real cataract data. We passed the real frames to all five models and generated the segmentations. Example predictions can be seen in Fig. 3. We observe that, despite being trained purely on simulated data, P2P was able to perform successful detection for some tools. For example, P2P-10K was able to segment correctly the retractors in column three (the lower part of the corresponding segmentation image). In the other columns, both P2P models distinguished the major parts of the tools from the background, despite assigning the wrong class. Specifically, in column three both models have created a fine segmentation of the tool in the upper left corner (also zoomed on the right). On the other hand, despite FCN-VGG having high performance on the simulated set, it was not able to generalise on the real set and it only produced a few detection (*e.g.* fourth column).

Using the binary tool presence annotation that was available in the real cataract dataset we measured the mean precision and mean recall of P2P-400 and P2P-10K on the real set. P2P-400 achieved 8 and 21% and P2P-10K achieved 7 and 28% mean precision and recall, respectively. The results of applying transfer learning on real data are encouraging because P2P was able to distinguish tools from the background and in many cases it created fine segmentations.

From the evaluation of our models it is highlighted that FCN-VGG performs better when trained on higher resolution images (FCN-VGG-10K-Large). When lowering the image resolution the objects can be distorted and lose important information. This poses an extra challenge detecting objects with CNNs. However, we can see that despite P2P being trained on lower resolution, it had a comparable performance on the simulated dataset and outperformed FCN-VGG on transfer learning. Therefore, we conclude that the domain differences between the simulation and reality pose a larger challenge in transfer learning than the resolution of the training images. Of course, using a larger training set could potentially increase the performance of P2P and is an experiment that we will be exploring in our future work.

An additional challenge that is highlighted in the mean accuracy and mean IU in our experiments is class imbalance. Indeed, within our training set some of the classes have more instances than others. For example, a capsulorhexis cystotome has >2000 instances whereas a primary incision knife has <300 instances. To balance the training classes we decided on a global threshold of 1134 instances per class and under-sampled the dataset. We discarded classes with fewer instances and randomly sampled the remaining ones based on the inverse of class frequency. Hence we constructed a new training simulated set of seven classes (six tools and one background class) for a total of 5851 training frames. Similarly we gathered a new validation and test set of 1955 and 1845 frames, respectively. We trained FCN-VGG and P2P on the new balanced dataset on high and low resolution (960×540 and 256×256), respectively, and called these models FCN-VGG-Large-balanced and P2P-balanced. From the evaluation of both models on the simulated domain we noticed an increase in performance highlighted in the new mean accuracy and mean IU shown in Table 3. We observe that both FCN-VGG-Large-balanced and P2P-balanced have a significant improvement in the class-oriented metrics (mean accuracy and mean IU), respectively, hence we conclude

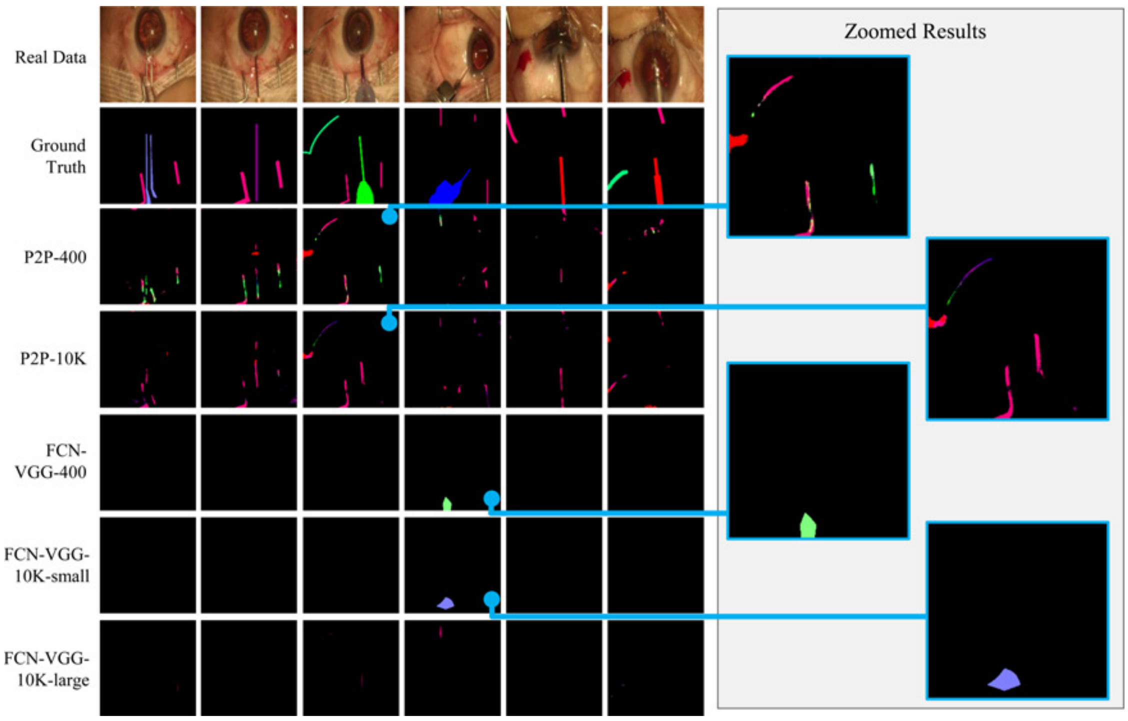

Fig. 3 *Experimental results presenting all the outputs of the two learning models for different parameters (training dataset number and resolution) applied on a real dataset. The real images are shown in the first row, while zoomed image results are presented on the right. We have manually labelled the ground-truth segmentations of the images in the second row*

Table 3 Evaluation metrics for the comparison between the balanced and imbalanced models on the simulated domain

Model	Pixel accuracy	Mean accuracy	Mean IU	fwIU
FCN-VGG-10K-Large	0.977	0.639 ± 0.322	0.526 ± 0.333	0.958
P2P-10K	0.982	0.503 ± 0.363	0.260 ± 0.350	0.974
FCN-VGG-10K-Large-balanced	0.972	0.709 ± 0.276	0.531 ± 0.330	0.950
P2P-balanced	0.985	0.527 ± 0.452	0.432 ± 0.400	0.974

that balancing the classes had a positive effect on the performance of our models. In addition, similar to the first experiments, FCN-VGG achieves higher class-oriented metrics, suggesting it is able to retrieve more classes successfully than P2P. Finally, we evaluated the balanced models on the real set. P2P-balanced exhibited performance similar to P2P-400 and P2P-10K achieving a precision and recall of 7 and 13%, respectively. FCN-VGG-Large-balanced, on the other hand, exhibited an increased performance comparing all other FCN-VGG models and can be compared with P2P. Both the precision and recall of FCN-VGG-Large-balanced are 11%.

We should note that although the new training set is balanced, lowering the number of classes from 14 to 7 leads to a slightly different experiment and we cannot directly compare with the original one. Another way to address class imbalance would be to use a loss function that calculates the weighted sum of the losses of each individual class based on the class frequency [23]. Alternatively, we could apply stratification techniques to make sure all classes are parsed when training the model [24]. Although these methods were not explored in the Letter we will be exploring their application to address class imbalance in future works.

4. Discussion and conclusions: In this Letter, we focused on showing that a simulated surgical video can potentially be used as an input to training deep learning architectures for vision-based surgical instrument detection and segmentation. To the best of our knowledge, this is the first attempt to perform object detection by unsupervised domain adaptation and, also the first attempt to generalise on real surgical data while training on purely synthetic. We tested two popular deep learning methodologies using CNNs with transfer learning to adapt simulation trained models to real data, and using GANs to perform style transfer as a means for solving the labelling problem. Our results for CNN transfer learning indicate that additional efforts are needed to properly adapt the model from the input to the output domain. Surprisingly, our results with the GAN approach are promising and instruments are detected well. Despite promising detection, the classification of different tools requires further work and improvement. It is also worth mentioning that our approach is feasible for segmenting/classifying anatomical structures. Since our training dataset is being generated by a simulator, we can potentially generate anatomical structures along with the corresponding labels. However, the size of this dataset is necessary to be significantly larger due to the fact that anatomical structures have much more variation in appearance. We should also notice that the CATARACTS grand challenge is still on-going and therefore the results of this work still cannot be compared with results from other participants. We believe this preliminary study shows a promising direction for further exploration of employing deep learning methods for CAS systems without being impeded by the lack of large labelled datasets which are typically the cornerstone of deep learning in computer vision.

5. Acknowledgements: We thank Darren Holdaway for the infrastructure support, Luy Spurge and Robert Joosten for rendering the simulated data, and Dr Costas Fantis for identifying the common tools between the synthetic and real datasets.

6. Funding and declaration of interests: The authors report a patent Surgical Simulation for Training Detection and Classification Neural Networks pending to Kinosis Ltd., London, United Kingdom. Patent Number: US 62/519,084.

7 References

[1] Maier-Hein L., Vedula S., Speidel S., ET AL.: 'Surgical data science: enabling next-generation surgery', arXiv:1701.06482, 2017
[2] Weiser T.G., Haynes A.B., Lashoher A., ET AL.: 'Perspectives in quality: designing the WHO surgical safety checklist', *Int. J. Qual. Health Care*, 2010, **22**, (5), pp. 365–370
[3] Vedula S.S., Ishii M., Hager G.D.: 'Objective assessment of surgical technical skill and competency in the operating room', *Annu. Rev. Biomed. Eng.*, 2017, **19**, (1), pp. 301–325
[4] Padoy N., Blum T., Ahmadi S.A., ET AL.: 'Statistical modeling and recognition of surgical workflow', *Med. Image Anal.*, 2012, **16**, (3), pp. 632–641
[5] Dosis A., Aggarwal R., Bello F., ET AL.: 'Synchronized video and motion analysis for the assessment of procedures in the operating theater', *Arch. Surg.*, 2005, **140**, (3), pp. 293–299
[6] Katić D., Wekerle A.L., Gärtner F., ET AL.: 'Knowledge-driven formalization of laparoscopic surgeries for rule-based intraoperative context-aware assistance'. Information Processing in Computer-Assisted Interventions (IPCAI), 2014 (LNCS, **8498**), pp. 158–167
[7] Zappella L., Béjar B., Hager G., ET AL.: 'Surgical gesture classification from video and kinematic data', *Med. Image Anal.*, 2013, **17**, (7), pp. 732–745
[8] Rieke N., Tan D.J., Alsheakhali M., ET AL.: 'Surgical tool tracking and pose estimation in retinal microsurgery'. Medical Image Computing and Computer-Assisted Intervention, 2015 (LNCS, **9349**), pp. 266–273
[9] Du X., Allan M., Dore A., ET AL.: 'Combined 2D and 3D tracking of surgical instruments for minimally invasive and robotic-assisted surgery', *Int. J. Comput. Assist. Radiol. Surg.*, 2016, **11**, (6), pp. 1109–1119
[10] DiPietro R., Lea C., Malpani A., ET AL.: 'Recognizing surgical activities with recurrent neural networks', Medical Image Computing and Computer-Assisted Intervention, 2016 (LNCS, **9900**), pp. 551–558
[11] Lin H.C., Shafran I., Murphy T.E., ET AL.: 'Automatic detection and segmentation of robot-assisted surgical motions', *Med. Image Comput. Comput. Assist. Interv.*, 2005, **8**, (Pt 1), pp. 802–810
[12] Bouget D., Allan M., Stoyanov D., ET AL.: 'Vision-based and marker-less surgical tool detection and tracking: a review of the literature', *Med. Image Anal.*, 2017, **35**, pp. 633–654
[13] Krizhevsky A., Sutskever I., Hinton G.E.: 'Imagenet classification with deep convolutional neural networks'. Advances in Neural Information Processing Systems (NIPS), 2012, pp. 1097–1105
[14] Long J., Shelhamer E., Darrell T.: 'Fully convolutional networks for semantic segmentation'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Boston, October 2015, pp. 3431–3440
[15] Russakovsky O., Deng J., Su H., ET AL.: 'Imagenet large scale visual recognition challenge', *Int. J. Comput. Vis.*, 2015, **115**, (3), pp. 211–252
[16] Girshick R., Donahue J., Darrell T., ET AL.: 'Rich feature hierarchies for accurate object detection and semantic segmentation'. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR), Columbus, June 2014, pp. 580–587

[17] Twinanda A.P., Shehata S., Mutter D., ET AL.: 'Endonet: a deep architecture for recognition tasks on laparoscopic videos', *IEEE Trans. Med. Imaging*, 2017, **36**, (1), pp. 86–97

[18] Isola P., Zhu J.-Y., Zhou T., ET AL.: 'Image-to-image translation with conditional adversarial networks'. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR), Hawaii, July 2017

[19] Sugand K., Mawkin M., Gupte C.: 'Validating touch surgery™: a cognitive task simulation and rehearsal app. for intramedullary femoral nailing', *Injury*, 2015, **46**, (11), pp. 2212–2216

[20] Simonyan K., Zisserman A.: 'Very deep convolutional networks for large-scale image recognition'. Int. Conf. Learning Representations (ICRL), San Diego, May 2015, pp. 1–14

[21] Everingham M., Van Gool L., Williams C.K.I., ET AL.: 'The PASCAL visual object classes challenge 2011 (VOC2011) results', *Int. J. Comput. Vis.*, 2011, **88**, (2), pp. 303–338

[22] Jia Y., Shelhamer E., Donahue J., ET AL.: 'Caffe: convolutional architecture for fast feature embedding'. Proc. ACM Int. Conf. Multimedia (ACM), Orlando, November 2014, pp. 675–678

[23] Eigen D., Fergus R.: 'Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture'. Proc. Int. Conf. Computer Vision (ICCV), Santiago, December 2015, pp. 2650–2658

[24] Manish S., Anirban M., Angelika S., ET AL.: 'Addressing multi-label imbalance problem of surgical tool detection using CNN', *Int. J. Comput. Assist. Radiol. Surg.*, 2017, **12**, (6), pp. 1013–1020