Procedurally Generated Colonoscopy and Laparoscopy Data For Improved Model Training Performance

Thomas Dowrick[1], Long Chen[1], Joao Ramalhinho[1], Juana González-Bueno Puyal[1], and Matthew J. Clarkson[1]

Wellcome EPSRC Centre for Interventional and Surgical Sciences, UCL

Abstract. The use of synthetic/simulated data can greatly improve model training performance, especially in areas such as image guided surgery, where real training data can be difficult to obtain, or of limited size. Procedural generation of data allows for large datasets to be rapidly generated and automatically labelled, while also randomising relevant parameters within the simulation to provide a wide variation in models and textures used in the scene.
A method for procedural generation of both textures and geometry for IGS data is presented, using Blender Shader Graphs and Geometry Nodes, with synthetic datasets used to pre-train models for polyp detection (YoloV7) and organ segmentation (UNet), with performance evaluated on open-source datasets.
Pre-training models with synthetic data significantly improves both model performance and generalisability (i.e. performance when evaluated on other datasets). Mean DICE score across all models for liver segmentation increased by 15% (p=0.02) after pre-training on synthetic data. For polyp detection, Precision increased by 11% (p=0.002), Recall by 9% (p=0.01), mAP@.5 by 10% (p=0.01) and mAP@[.5:95] by 8% (p-0.003).
All synthetic data, as well as examples of different Shader Graph/Geometry Node operations can be downloaded at https://doi.org/10.5522/04/23843904.

Keywords: Simulation · Image Guided Surgery · Data Engineering.

1 Introduction

A majority of researchers in Image Guided Surgery (IGS) are involved with machine learning in some form (registration, segmentation, stereo reconstruction, classification etc). However, the lack of application specific training data is a major blocker for development. In addition, the time-consuming process of manually labelling data is especially challenging for medical data, as labelling complex intraoperative scenes or radiological data typically requires the intervention of a trained clinician.
The wider computer vision community has benefited from large open-sourced

Fig. 1. Rendered synthetic data for Laparoscopy (left column) and Colonoscopy.

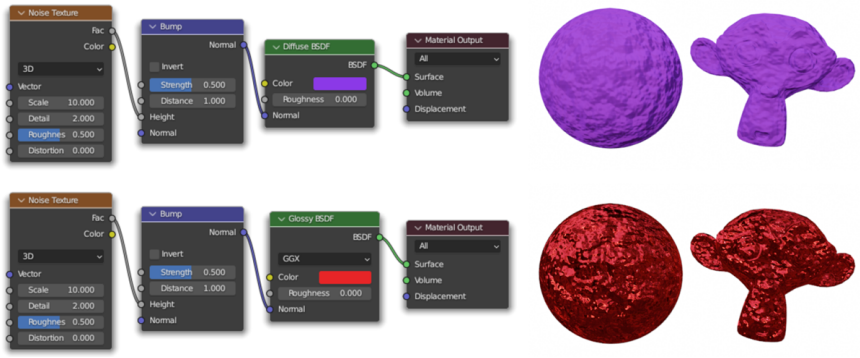

Fig. 2. Example of Blender Shader Graph for basic shading and normal maps. By combining different nodes, and adjusting their parameters, different textures/effects can be generated.

datasets, including both real (e.g. ImageNet, KITTI) and simulated (e.g. Scene Flow, Virtual KITTI). While synthetic data can underperform in training when compared to real data, due to the so called 'domain gap', recent advances in domain specific simulation have produced models with equivalent, or in some cases superior performance to those trained only on real data [10][16][13]. Synthetic data has the additional advantage of being able to generate more accurate and complex labels, without the time/cost overheads of manual labelling.
IGS researchers have applied several methods from the machine learning community to generate synthetic data [16][11][4][14][9][15][6]. However, procedural generation, a method of creating data algorithmically from a pre-defined set of rules, is not an area that has had much investigation in IGS, despite showing promising results on conventional image recognition tasks [13]. As they allow control over the entire scene, procedural methods can also be integrated into active learning/active simulation pipelines, where the simulation parameters are updated on the fly in response to the network performance.

1.1 Contribution

In this work, methods for the generation of high quality, procedurally rendered data for IGS applications are described. This includes a fully procedural generation method, with no user inputs required, for generating colonoscopy data for polyp detection (which the authors believe to be a first), and a partially procedural method, where anatomically accurate models are used, with procedural textures, for liver segmentation during laparoscopic liver surgery.
Data was rendered using Blender (https://www.blender.org) (Figure 1), taking ad-vantage of two main areas of functionality. The first is the use of Shader Graphs to generate realistic tissue textures. The second is Geometry Nodes, which allows for the entire geometry of the scene to be defined, and modified procedurally. On top of this, custom scripting allows for randomization of rele-

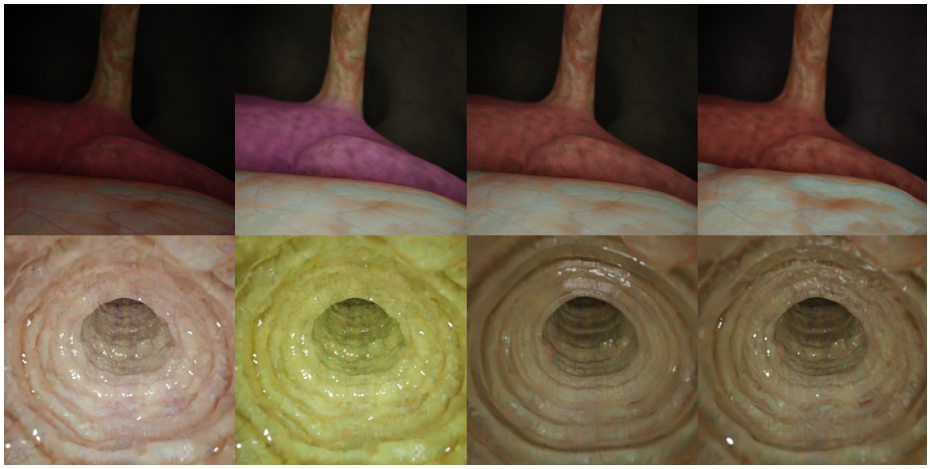

Fig. 3. Randomising texture and lighting parameters on laparoscopy (above) and colonos- copy (below) data.

vant parameters within the scene, allowing large, varied datasets to be rapidly generated.
The use of this data for model pre-training boosts performance, when evaluated on a number of publicly available datasets, in both laparoscopy and colonoscopy. Data used for training, along with original Blender files, to allow for data replication, is available for download (https://doi.org/10.5522/04/23843904).

2 Methods

2.1 Shader Graphs for Texture Generation

The use of visual editors and node based approaches to producing shaders has increased in recent years, providing a layer of abstraction above shader code (HLSL, OSL etc), allow the user to design shaders more intuitively, with instant feedback as parameters are changed. All major 3D graphics tools now include this functionality, including Unreal (Material Editor), Unity (Shader Graph), Houdini (Materials) and Blender. (Shader Graph) An example Shader Graph in Blender is shown in Figure 2, making use of the following nodes:

- Noise Texture: this node generates a procedural noise pattern, often used for creating natural-looking textures or adding surface imperfections. It offers various parameters to control the type, scale, and intensity of the noise pattern.
- Bump: this node perturbs the surface normals of a material, simulating surface details without actually modifying the geometry.
- Diffuse BSDF: The Diffuse BSDF node represents a Lambertian diffuse material.

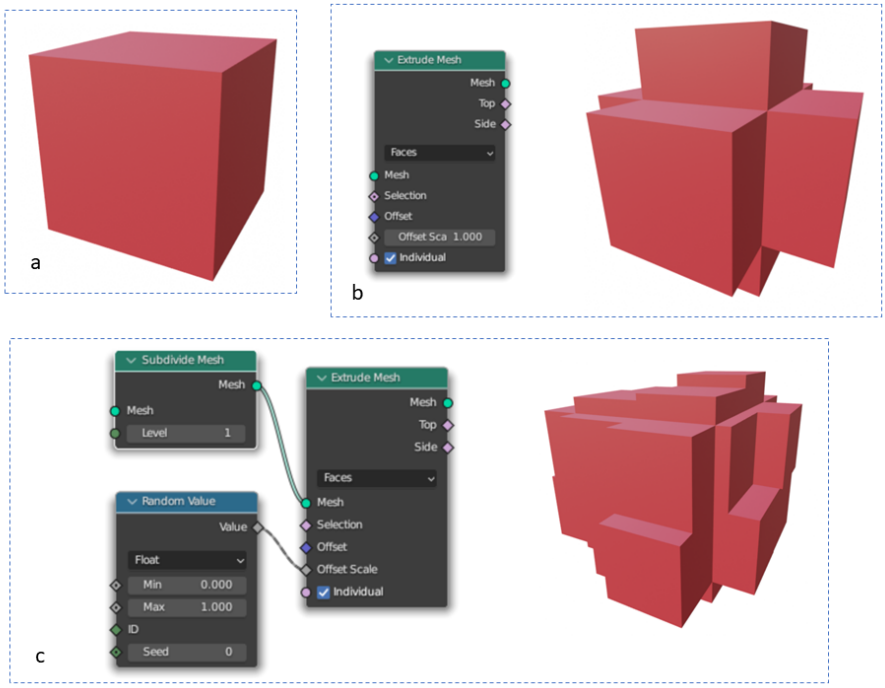

Fig. 4. Geometry Nodes example. (a) Input Cube. (b) Linear extrusion of each face. (c) Subdivision of the mesh, then extrusion of each face to a random distance. Output geometry data is generated on the fly from as the input geometry/parameters are changed.

– Material Output: The Material Output node is the final node in the shader graph and serves as the endpoint for the material. It combines different shader outputs, such as Diffuse BSDF and links them to the surface of the 3D model for rendering.

The approach used in this work was to generate all textures procedurally, making use of more than 20 different Shader Graph nodes, allowing for fine grained control over all aspects of the texture's appearance, including albedo, bump mapping, displacement, subsur-face scattering, reflectance, glossiness etc.
Custom Shader Graphs were created for each organ, to match the properties of the real tissues as closely as possible. Within each graph, key parameters were identified which were to be randomly varied (Figure 3) at simulation time, as well as the ranges over which to randomise. The appearance of each Shader Graph was manually tuned, and the appearance compared visually to sample images of each target tissue. All textures used for training in this work were generated using a Shader Graph, and these can be found in the accompanying dataset release (https://doi.org/10.5522/04/23843904).

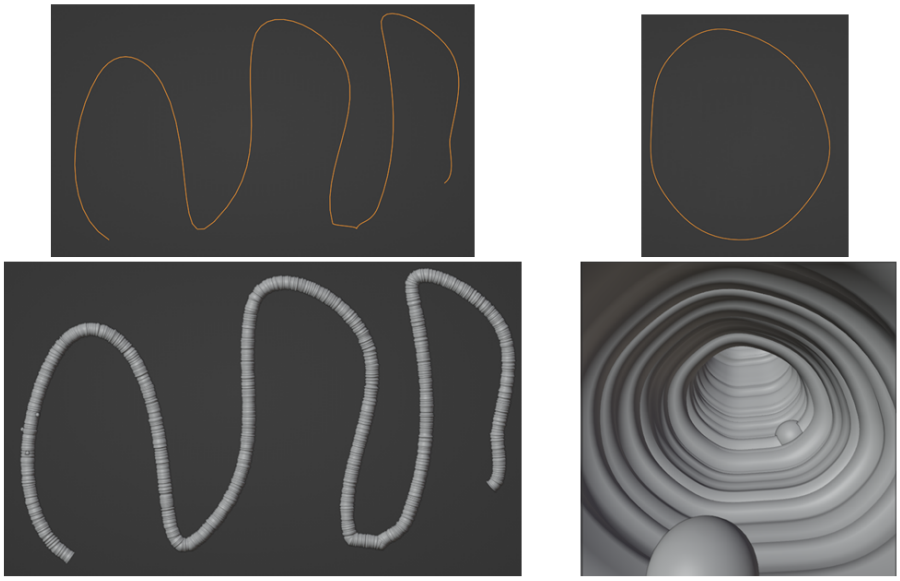

Fig. 5. Procedural generation of colon model. Clockwise from top left - curve used to define shape of colon; curve used to define cross section of colon; internal view of generated colon (no shading) with randomly placed polyps; external view of generated colon. The colon shape will update in real time in response to changes to either of the input curves. Further customization is carried out through randomization of Geometry Node parameters at simula- tion time.

2.2 Geometry Nodes for Model Generation

Procedural modeling is a powerful approach in computer graphics and 3D design that allows for the automatic generation of complex shapes, textures, and animations using algorithms and rules. Instead of manually creating each element, procedural modeling relies on mathematical functions, parameters, and logical operations to define the geometry and appearance of objects. This approach offers numerous advantages, including scalability, flexibility, and the ability to create variations easily. Procedural models can be modified parametrically, enabling quick adjustments without redoing the entire design. As a result, procedural modeling is widely used in various industries, including video games, visual effects, architectural visualization, and simulation. Here, we use Blender's Geometry Nodes feature (Figure 4) which enables procedural modeling by connecting nodes that manipulate input geometry, perform operations like transformations and deformations, and generate or modify mesh topology. Attribute and math nodes manage data and perform calculations, while input nodes provide user-defined parameters. Geometry nodes are used to procedurally generate both a colon model, and to distribute polyps across the surface of the colon. Start-

ing with a single curve to represent the shape of the colon, the entire model, including the location and size of polyps, is generated from scratch (Figure 5).

2.3 Rendering/Synthetic Dataset Generation

For each frame of data, texture parameters were randomised controlling displacement magnitude, bump map magnitude, colour of the organs/tissues, subsurface scattering parameters, noise levels etc. The intensity of the lighting, the level of motion blur, and the position, look direction and focal distance of the camera were also adjusted. All images were rendered using the Cycle raytracing engine, at dimensions of 512x512, with a noise threshold of 0.1 and 256 samples. Render time for each frame was $\sim$ 5s.

Colonoscopy – 50,000 frames Geometry Nodes parameters, controlling the shape of the colon, and the position, size and distribution of polyps were randomized per frame.

Laparoscopic liver surgery – 50,000 frames Publicly available models for liver, gallbladder, etc. were used, and mesh primitives were used to represent other organs/tissues where appropriate (abdominal cavity = sphere etc.).

Custom Labelling Semantic segmentations (Liver) were acquired by re-rendering the scene with each object assigned a flat colour, with bounding boxes (Polyps) derived from the minimum/maximum extents of the segmentation information.

Table 1. Data split for colonoscopy data. Brackets indicate the number of labeled polyps in that set.

	Total Images	**Train**	**Validation**	**Test**
Kansas	37899	28773 (27048)	4254 (4214)	4872 (4719)
HyperKvasir	1000	800 (972)	100 (121)	100 (113)
LD	4-186	20855 (18900)	3934 (4569)	15397 (15268)
PolypGen	1471	1178 (1191)	88 (98)	208 (204)
Blender	50000	45000 (60827)	5000 (7685)	N/A

Table 2. Data split for laparoscopy data. Brackets indicate the number of distinct patients/procedures in that set.

	Total Images	**Train**	**Validation**	**Test**
DSAD	1430 (23)	1131 (18)	101 (2)	119 (3)
CholecSeg8k	8080 (19)	6080 (15)	1000 (2)	2000 (2)
SISVSE	4510 (40)	3588 (32)	457 (4)	462 (4)
Blender	50000	45000	5000	N/A

2.4 Evaluation Datasets

Colonoscopy Four datasets were used: Kansas Polyp Dataset [7], HyperKvasir [2], LDPolypVideo [8] and PolypGen [1]. Only polyp detection was considered, so from each data set, the relevant subset of data was used (HyperKvasir for example also contains upper GI tract data). Labels were converted to the COCO format. Kansas and LD datasets have a train/test/validation split already defined, which was left unchanged. LD data was split 80/10/10, and PolypGen was randomly split into 5 patients for train, 1 for test and 1 for validation (Table 1).

Laparoscopic liver segmentation Three datasets were used: Dresden Surgical Anatomy Dataset (DSAD) [3], CholecSeg8k [5], SISVSE[16]. DSAD contains both single organ labelling and multi-organ labelling datasets. For this work, only the data from the liver single organ subset was used. The full SISVSE and CholecSeg8k datasets were used, with any non-liver labels removed.

Each dataset provides data from a number of separate patients/procedures. Data was randomly split into training, test and validation data, with an approximate 80/10/10 split of images between the three (Table 2). Actual splits deviate slightly from this, as the number of images for each patient varies. It should be noted that while CholecSeg8k has the highest number of images, the dataset consists of multiple sets of sequential frames taken from the same procedures, whereas each frame in DSAD and SISVSE are non-sequential/from different procedures.

2.5 Model Training

Laparoscopy – semantic segmentation Semantic segmentation was evaluated using a standard UNet configuration, with combined DICE loss and Cross Entropy loss, RMSprop optimizer, and learning rate of 1e-5. The network was trained on each dataset individually, as well as with pre training on Blender data for each dataset. Pre-training on Blender data was for 10 epochs; all other training runs were 50 epochs. This resulted in 6 trained models (each dataset with and without Blender pre-training), each of which was evaluated on the three sets of test data, with the DICE score for liver classification recorded.

Colonoscopy – polyp bounding box detection Polyp detection was trained using Yolov7[12]. Default training parameters were used for the full YoloV7 network, with pre-trained ImageNet weights loaded. A model was trained for 100 epochs on each of the 4 datasets, as well as being pre-trained on the Blender data and post trained on each dataset. This resulted in 8 trained models (each dataset with and without Blender pre-training), each of which was evaluated on the four sets of test data.
The metrics reported by YoloV7 are precision, recall and mean average precision (mAP). mAP is calculated both for a single IoU of 0.5 (mAP@.5), and as an average of the mAP for IoU values between 0.5 and 0.95 (mAP@[.5:.95]).

3 Results

For laparoscopy data (Table 3), the use of synthetic data for pre-training increased the DICE score in 8 out of 9 cases, with the average change being an increase of 15% (p=0.02, using paired t-test).
For colonoscopy data (Table 4), for each evaluation metric, for each dataset, the highest value was achieved when the synthetic data was used for pre-training (cells with shaded background). If the training dataset is excluded from evaluation, 11 out of 16 metrics are achieved on pre-trained data (bold text in table); 3 are unchanged, and 2 are lower following pre-training. When the performance of individual metrics is compared with/without pre-training, then Precision is increased 12/16 times (Average change +11%, p=0.01), Recall 14/16 (+9%, p=0.002), MAP@.5 13/16 (+10%, p=0.01), MAP@[.5:.95] 13/16 (+8%, p=0.003).

Table 3. Liver segmentation DICE score, out of 100. Rows indicate training dataset, columns the test dataset. Cells with highlighted background show the highest value for that metric, across all models. Bold values indicate the highest value when the dataset used for training is excluded (e.g. excluding Cholec trained models from evaluation on Cholec data)

	Cholec	**DSAD**	**SISVSE**
Cholec	75	74	73
DSAD	37	85	61
SISVSE	23	77	77
Blender + Cholec	79	87	78
Blender + DSAD	55	96	**79**
Blender + SISVSE	**71**	**92**	91

Table 4. Polyp detection results. All values given as a score out of 100. Columns represent the results on the test sets, and rows are the different trained models. Cells with a highlighted background show the highest value for that metric, across all models. Bold values indicate the highest value when the dataset used for training is excluded (e.g. excluding HyperKvasir models from evaluation on HyperKvasir test set). B = Blender, KA = Kansas, KV = Kvasir, L = LD, P = PolypGen.

	Kansas (KA)				**Kvasir (KV)**				**LD (L)**				**PolypGen (P)**			
	P	R	mAP .5	mAP [.5:.95]	P	R	mAP .5	mAP [.5:.95]	P	R	mAP .5	mAP [.5:.95]	P	R	mAP .5	mAP [.5:.95]
KA	83	74	82	49	**86**	54	61	40	55	36	37	17	45	33	33	16
KV	46	23	23	12	82	66	73	44	37	15	14	07	20	28	16	08
LD	64	38	41	24	84	67	74	46	**69**	47	52	24	54	43	44	21
P	**81**	39	50	**31**	**86**	74	80	55	52	34	36	18	69	50	59	37
B+KA	88	83	92	58	80	62	66	44	62	**42**	**45**	**21**	59	45	43	24
B+KV	65	41	44	28	93	80	84	64	60	38	41	20	**74**	**52**	**63**	**39**
B+LD	68	29	33	20	83	65	73	44	73	48	55	26	59	44	45	25
B+P	74	**43**	**50**	30	**86**	**81**	**83**	**62**	63	41	**45**	**21**	71	66	67	45

4 Discussion

For both the laparoscopy (Table 3) and colonoscopy (Table 4) datasets, the use of synthetic data improved model performance, across all metrics, compared with train-ing only on real data. The results given in this work show that the method employed for procedural generation of training data can be used to improve model performance. It is envisaged that such methods would be complementary to existing approaches for data synthesis (GANs, diffusion models etc.) either by the use of multiple sources of synthetic data for training, or for example, by generating target geometries and labels using Geometry Nodes, and then applying an alternative method for texture synthesis. Being able to generate synthetic data in this way also extends the use of synthetic data to areas where there may not be sufficiently large training datasets to utilize deep learning methods. Further work is underway to consider the effects of changing the ratio of synthetic to real data when training, and to make use of Geometry Nodes to provide more fine-grained labels, such as polyp sizing, for more advanced applications.

Acknowledgements This work was funded by EPSRC (EP/V052438/1), and supported by a Microsoft Azure Research Grant, and an Oracle Cloud Computing Grant.

References

1. Ali, S., Jha, D., Ghatwary, N., Realdon, S., Cannizzaro, R., Salem, O.E., Lamarque, D., Daul, C., Riegler, M.A., Anonsen, K.V., Petlund, A., Halvorsen, P., Rittscher, J., de Lange, T., East, J.E.: Polypgen: A multi-center polyp detection and segmentation dataset for generalisability assessment (6 2021)
2. Borgli, H., Thambawita, V., Smedsrud, P.H., Hicks, S., Jha, D., Eskeland, S.L., Randel, K.R., Pogorelov, K., Lux, M., Nguyen, D.T.D., Johansen, D., Griwodz, C., Stensland, H.K., Garcia-Ceja, E., Schmidt, P.T., Hammer, H.L., Riegler, M.A., Halvorsen, P., de Lange, T.: Hyperkvasir, a comprehensive multi-class image and video dataset for gastrointestinal endoscopy. Scientific Data **7** (2020). https://doi.org/10.1038/s41597-020-00622-y
3. Carstens, M., Rinner, F.M., Bodenstedt, S., Jenke, A.C., Weitz, J., Distler, M., Speidel, S., Kolbinger, F.R.: The dresden surgical anatomy dataset for abdominal organ segmentation in surgical data science. Scientific Data **10**, 3 (1 2023). https://doi.org/10.1038/s41597-022-01719-2
4. Funke, I., Robu, M., Bodenstedt, S., Strenger, L., Engelhardt, S., Clarkson, M., Gurusamy, K., Davidson, B., Maier-Hein, L., Riediger, C., Welsch, T., Weitz, J., Speidel, S., Pfeiffer, M.: Generating large labeled data sets for laparoscopic image processing tasks using unpaired image-to-image translation. CoRR (2019)
5. Hong, W.Y., Kao, C.L., Kuo, Y.H., Wang, J.R., Chang, W.L., Shih, C.S.: Cholecseg8k: A semantic segmentation dataset for laparoscopic cholecystectomy based on cholec80 (12 2020)
6. Jagtap, A.D., Heinrich, M., Himstedt, M.: Automatic generation of synthetic colonoscopy videos for domain randomization (5 2022)

7. Li, K., Fathan, M.I., Patel, K., Zhang, T., Zhong, C., Bansal, A., Rastogi, A., Wang, J.S., Wang, G.: Colonoscopy polyp detection and classification: Dataset creation and comparative evaluations. PLOS ONE **16**, e0255809 (8 2021). https://doi.org/10.1371/journal.pone.0255809
8. Ma, Y., Chen, X., Cheng, K., Li, Y., Sun, B.: LDPolypVideo Benchmark: A Large-Scale Colonoscopy Video Dataset of Diverse Polyps, pp. 387–396 (2021). https://doi.org/10.1007/978-3-030-87240-3_37
9. Moreu, E., McGuinness, K., O'Connor, N.E.: Synthetic data for unsupervised polyp segmentation (2 2022)
10. Olivier Pauly, Hauke Heibel, M.M.M.B.S.H.: An annotation saved is an annotation earned: Using fully synthetic training for object instance detection. Computer Vision and Pattern Recognition (2019)
11. Rivoir, D., Pfeiffer, M., Docea, R., Kolbinger, F., Riediger, C., Weitz, J., Speidel, S.: Long-term temporally consistent unpaired video translation from simulated surgical 3d data (2021). https://doi.org/10.1109/ICCV48922.2021.00333
12. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors (7 2022)
13. Wood, E., Baltrušaitis, T., Hewitt, C., Dziadzio, S., Cashman, T.J., Shotton, J.: Fake it till you make it: face analysis in the wild using synthetic data alone (2021). https://doi.org/10.1109/ICCV48922.2021.00366
14. Yonghao Long, Siu Hin Fan, Q.D.Y.W.: Neural rendering for stereo 3d reconstruction of deformable tissues in robotic surgery
15. Yoon, D., Kong, H.J., Kim, B.S., Cho, W.S., Lee, J.C., Cho, M., Lim, M.H., Yang, S.Y., Lim, S.H., Lee, J., Song, J.H., Chung, G.E., Choi, J.M., Kang, H.Y., Bae, J.H., Kim, S.: Colonoscopic image synthesis with generative adversarial network for enhanced detection of sessile serrated lesions using convolutional neural network. Scientific Reports **12**, 261 (1 2022). https://doi.org/10.1038/s41598-021-04247-y
16. Yoon, J., Hong, S., Hong, S., Lee, J., Shin, S., Park, B., Sung, N., Yu, H., Kim, S., Park, S., Hyung, W.J., Choi, M.K.: Surgical Scene Segmentation Using Semantic Image Synthesis with a Virtual Surgery Environment, pp. 551–561 (2022). https://doi.org/10.1007/978-3-031-16449-1_53