Estimation of endoscope camera motion trajectories from surgical videos with Deep Learning

Andrea Naclerio
Person code: 10934883
andrea.naclerio@mail.polimi.it

Ali Shadman Yazdi
Person code: 10896277
ali.shadman@mail.polimi.it

Siavash Taleghani
Person code: 10775126
siavash.taleghani@mail.polimi.it

I. Introduction

Comprehending surgical scenes is crucial for the technological framework of upcoming intervention-assisting systems in endoscopic surgeries. Tracking the pose of the endoscope is a pivotal element in this process, yet it presents challenges attributable to factors such as varying illumination conditions, tissue deformation, and the rhythmic motion caused by organ breathing. In this scenario, the project aims to build a model that estimates the camera pose between 2 consecutive frames in order to predict the overall camera trajectory.

A State of the art

A.1 Machine learning methods

Various methods have been introduced for the estimation of camera pose in endoscopy videos or image sequences. This process proves instrumental in identifying the location of lesions and reconstructing a color-textured 3D model of organs such as the stomach or other body parts under examination [1] . However, computer vision tasks and processing using endoscopic videos or image sequences pose unique challenges due to the texture-less nature of the human organs as captured by endoscopic cameras. The absence of distinct textures can result in significant illumination changes and specular reflections during image acquisition, also known as Lambertian reflectance, leading to complexities in image processing, Numerous studies have been conducted to address and mitigate these challenges, aiming to develop effective solutions for accurate camera pose estimation in the context of endoscopy.

In the study conducted by Trinh et al. [2], an optical flow method was proposed, incorporating the detection and removal of specular reflections. This approach aimed to develop an image mosaicing method, addressing the challenges posed by strong illumination changes and specular reflections. Endoscopic image mosaicing, facilitated by such techniques, holds potential for diagnosing cancerous lesions and inflammations [3] Optical flow, a fundamental concept in computer vision, characterizes the displacement or movement of objects in images or image sequences over time. It finds applications in tasks such as motion tracking and object recognition. A typical pipeline for image mosaicking comprises preprocessing, registration, image stitching, and post-processing steps such as color correction [2]. Among these, registration stands out as a crucial step, and its effectiveness hinges on the characteristics of the input images. In the case of endoscopic images, which lack texture and exhibit strong illumination variations, achieving accurate registration becomes challenging. This study proposed illumination-invariant descriptors to address such challenges. The concept behind illumination-invariant descriptors lies in identifying pixels with luminance components greater than their chromatic luminance [4]. This technique can serve as a noise reduction approach in conjunction with other image processing techniques such as Scale-Invariant Feature Transform (SIFT) [5] where distinctive keypoints in images are identified, regardless of their scale or orientation. Or Simultaneous Localization and Mapping (SLAM) [6] where a device, such as a robot dynamically navigates and constructs a map of its surroundings in real-time, and various deep learning models [7], thereby enhancing the efficacy of tasks involving endoscopic videos or image sequences.

A.2 Deep learning methods

The proposed method by [9] is to tackle problems of pose estimation in minimally invasive surgery induced by difficult situations such as deformations caused by instruments and breathing, self or instrument-based occlusions, textureless surfaces and tissue specularities. The method integrates deep learning techniques, particularly a Deep Declarative Network (DDN), into the camera pose estimation process for endoscopic videos. The Deep Declarative Network is utilized to learn adaptive per-pixel weight mappings 2D (x) and 3D (x), which play a crucial role in balancing the contributions of different geometric losses based on the content of input images. By leveraging the expressiveness of neural networks, the method can effectively adapt the contribution of image regions to the pose estimation process.

The network architecture used for learning the weight mappings is a 3-layer [10] with a Sigmoid activation function to ensure outputs fall within the range [0, 1]. The input to these networks includes various elements used to compute residuals, such as image data, depth maps, optical flow, and other relevant parameters. During training, the network parameters are learned to minimize the supervised training loss, which compares the estimated pose with ground truth data.

$$L = \|p_t^* - p_t(gt)\|_1 \tag{1}$$

p_t^*: estimated pose - $p_t(gt)$:ground truth pose

The Deep Declarative Network enables end-to-end learning by facilitating implicit differentiation of the pose optimization process. This allows the network to compute gradients of the weight map parameters with respect to the training loss, even though the pose optimization itself is not directly differentiable. The optimization process aims to find a local or global minimum in the forward pass, making use of a Nonlinear Least-Squares (NLLS) problem.

Overall, the integration of deep learning aspects, particularly the use of Deep Declarative Networks, enhances the robustness and adaptability of the camera pose estimation method, making it more effective in challenging endoscopic surgical scenarios.

B Dataset

The dataset consists of 27 stereo-endoscopic RGB videos that were cropped to obtain only a single one. From each video, a variable number of frames were extracted together with the ground truth camera pose related to a fixed external reference frame.

II. Material and methods

A Pre-processing

The provided dataset comprises 27 endoscopic videos, each containing more frames than its predecessor. These videos capture data from fresh porcine cadaver abdominal anatomy, utilizing a da Vinci Xi endoscope and a projector to acquire high-quality depth maps of the scene [11]. The ground truth values are measured in millimeters, and invalid pixels are appropriately masked out. The ground truth for camera poses is represented by a 4x4 matrix relative to the first frame, incorporating stereo camera calibration specific to the sequence. Each frame of the video, converted into a sequence of images, is associated with a corresponding 4x4 matrix. This matrix indicates the position of the camera with respect to an external reference system. Each image in the image sequence has dimensions of 1280 pixels in width and 1024 pixels in height, featuring three channels that represent the colors red, green, and blue (RGB). To optimize computational efficiency and reduce model complexity, the images were resized by half, resulting in dimensions of 512x640. Furthermore, pixel values were normalized to fall within the range of 0 to 1. In an effort to enhance the performance of pose estimations, various preprocessing techniques were applied to the images. Figure 1(a) allustrates one of the endoscopic images without any preprocessing applied. Figure 1(b) presents the same image in grayscale. Grayscaling was employed to simplify the model, as the color of objects does not contribute essential information to the primary objective of camera pose prediction. The emphasis is primarily on capturing movements within the frames. Drawing inspiration from grayscale preprocessing, another technique was applied to further reduce complexity—binarization of the image, as depicted in Figure 1(c). The binarization process involves initial grayscaling, followed by specifying a threshold. Pixels with values below the threshold are set to 0, while those equal to or above the threshold are set to 1. The resulting image maintains a single channel, similar to grayscale, but with pixel values restricted to either 0 or 1. An additional preprocessing step involved the addition of grids to the images, as illustrated in Figure 1(d). The inclusion of grids serves to provide the model with a reference for assessing the extent of object movement within the image. In scenarios where the camera undergoes simultaneous zooming and rotation, incorporating grids aids the model in capturing these complex movements more effectively. In instances where transfer learning was applied, such as using the ResNet model, which mandates a 3-channel RGB input for each image, a predefined preprocessing step for ResNet is necessary. The preprocessing step zero-centers each color channel concerning the ImageNet dataset, without scaling. This is achieved by subtracting the mean value of each color channel from the images, ensuring that the resulting values are centered around zero.

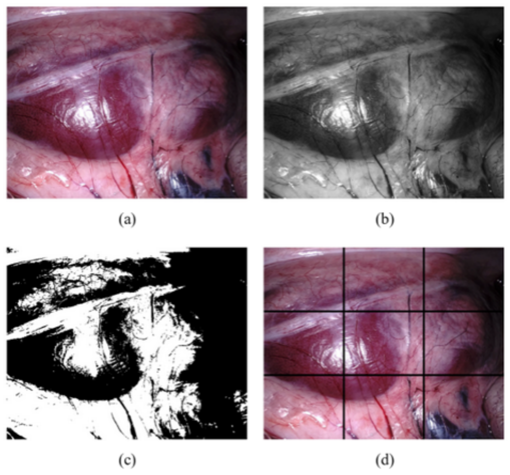

Fig. 1. Results of preprocessing methods on input images

The provided 4x4 transformation matrix for each image, in a sequence, is defined with respect to a global reference system. However, the primary goal of the model is to predict a transformation matrix (T) between the previous and the following frame. In other words, the camera pose on the following frame considering as reference the camera pose in the previoius one. To facilitate this objective, matrix preprocessing is necessary to transform the matrices into the preferred format. In order to accomplish that, the inverse of the transformation matrix (${}^{0}T_{n}^{-1}$) of the previous frame was multiplied by the transformation matrix of the following one (${}^{0}T_{n+1}$):

$$ {}^{n}T_{n+1} = {}^{0}T_{n}^{-1} * {}^{0}T_{n+1} \tag{2}$$

This process is systematically applied to all paired of consecutive frames in a sequence. For an image sequence with a length of N frames, it results in N-1 translation matrices. The flexibility to skip frames can also be introduced as a way of data augmentation.

Since the homogenous transformation matrix was composed by 16 values that were correlated between each other, only the

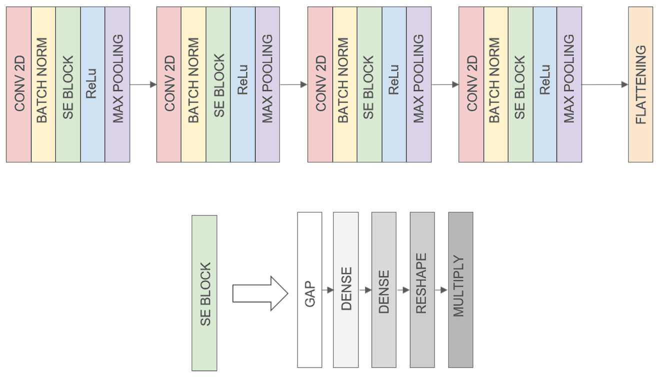

Fig. 2. Architecuture of the 'Scratch' Model

independent values were considered to reduce the complexity of the model output. For this reason, each 4x4 transformation matrix was decomposed into a 6-values vector, comprising 3 Euler angles to desribe the camera orientation and 3 coordinates (x,y,z) for the camera position. This decomposition involves separating the 3x3 rotation matrix and the translation vector from the original transformation matrix. To calculate the Euler angles, the singularity of the rotation matrix is checked, and the augmented arctangent function (ATan2) was employed to compute them.

Specifically, the angles were computed trough the following formulas, retrived by inverting the construction of the rotation matrix using the Euler angles, where r_{nm} denoted the different values of the rotation matrix, specifically n indicates the row and m the column:

$$\begin{aligned}
\beta &= ATan2\left(\pm\sqrt{r_{23}^2 + r_{33}^2},\, r_{13}\right) \\
\alpha &= ATan2\left(\frac{r_{33}}{cos(\beta)}, -\frac{r_{32}}{cos(\beta)}\right) \\
\gamma &= ATan2\left(\frac{r_{11}}{cos(\beta)}, -\frac{r_{12}}{cos(\beta)}\right)
\end{aligned} \tag{3}$$

Moreover, since there were always 2 possible solution for the β angle, the positive result was always considered in a consistent way.

As a result, the ground truth of the model is now represented by a vector with 6 elements, streamlining the information from the original 4x4 matrix.

B Models

In order to estimate the camera trajectory through the video's frames, it was necessary to assess how the camera moved from one frame to the other. To predict the overall camera path, the model was trained to manage a simpler task; takes as input two consecutive frames, one representing the starting point and the other the final one, estimating the final pose with respect to the initial one. Repeating this procedure for all the frames, it is possible to reconstruct the overall camera trajectory. To manage this task, it was needed to design properly a model that handles 2 images in parallel. So, the adopted model architecture was the so-called Siamese, that is composed of 2 identical branches that perform in the same way but on 2 different inputs. Each one receives as input a frame, an RGB image, that is processed through a combination of convolutional, activation, and pooling layers to extract relevant features.

In order to choose the model architecture that fitted the two branches, several models were tested. As first step, transfer learning models were adopted; more in detail, ResNet, and ResNet with squeeze and expansion blocks. Residual Networks, or ResNets, learn residual functions with reference to the layer inputs, instead of learning unreferenced functions. Instead of hoping each few stacked layers directly fit a desired underlying mapping, residual nets let these layers fit a residual mapping. They stack residual blocks ontop of each other to form network: e.g. a ResNet-50 has fifty layers using these blocks.Formally, denoting the desired underlying mapping as H(x), we let the stacked nonlinear layers fit another mapping of $F(x) = H(x) - x$. The original mapping is recast into $F(x) + x$.There is empirical evidence that these types of network are easier to optimize, and can gain accuracy from considerably increased depth.

The main limitation of using those architectures was the lack of flexibility in the model design due to the already fixed structure. To achieve better versatility in the selection of the layers and their combinations, it was decided to implement a model from scratch. The initial model fitted a base block that is repeated several times to create the overall structure; it was composed of 2D convolution (conv2D), activation, and max pooling (MP) layers. The conv2D is the standard layer used to extract specific patterns from the input image; it applies local kernel learned by the models during the training phase. The output generated by the layer is a simple linear combination of a local input region. So to avoid the multi-layers model acting as a simple linear combination of the inputs, an activation layer that fits a non-linear function, ReLu, is used. A max-pooling layer follows to reduce the image dimension and increase the receptive field of the next layers such that the kernels are spamming a bigger input region extracting higher-level features. Those architectures were applied in both branches to extract features from the two frames. After those models, a final flattering layer was inserted to transform the output volume in a 1D vector. So the two feature vectors extracted from the two frames, were concatenated and inputed to a sequence of fully dense layers. The last one was composed of 6 neurons that fitted a linear activation function since the predictions are 6 real numbers, 3 for the orientation and 3 for the position of the camera. Regarding the hidden dense layers, each one fitted a non-linear activation, specifically a tanh. Since the output values could be both positive and negative, this activation function was preferred rather than the ReLu that imposed only positive output. Moreover, each dense layer was followed by a dropout layer to prevent overfitting. Further structure was added to the standard structure trying to achieve better performance as through the squeeze and extension block.

C Results computation

In order to evaluate the model performance, graphical and numerical representations were computed.

The first was a 3D plot where the target camera trajectory and the predicted one were compared. As described in Section II.A, the model outputs and the targets were the camera pose of one frame with respect to the previous one. So the values were computed considering the previous frame as a reference system. Since in this plot the camera path is the final goal, only the camera position is relevant for the computation. To obtain the overall trajectory, each single 'local' camera position between 2 consecutive frames needs to be converted into a fixed external 'global' reference frame. So, the initial camera position in the first frame is considered and, at each step, the local position was summed up creating the complete camera trajectory.

The second representation is a 2D graph where the trajectory error was plotted. In particular, the absolute error between two corresponding points belonging from the target and predicted trajectory was shown. By doing so, it's possible to understand how the model was performing; despite the fact the predicted trajectory was not identical, it's possible to understand if the error increases during the path estimation or, as it would like to have, remain constant.

Additional numerical metrics were adopted both for the position and orientation in order to have a quantitative and objective evalutation of the model performance. Respectively, the Euclidean distance and the mean angular error were considered. More in detail, this last one was the result of the average between the angles creating by the predicted frame axes and the target one of the corresponding axes; in other words the angles created between the corresponding axes, e.g. target x-axis and the predicted one, were computed and averaged with the others.

D Experiements

As mentioned in section IIA, several preprocessing methods were tested and the one that achieved the best result is using the raw frames. So all the models were trained using the initial dataset, applying only a dowsample in the image dimension of a factor 2.

The 'scratch' model that achieved the best performance was the one that fitted an additional layer, the squeeze and expansion (SE) block. The concept behind it is related to the fact that some channels might be more important than others, so it makes sense to apply a weight to the channels based on their importance. The simplest way to do this is by scaling the most important channels by a higher value.

To generate the training set, frames belonging to the same video were coupled to each other considering two consecutive frames, previous and following. The model was trained on GPUs and due to the limitation of the RAM memory, only 3 videos were used for the training and validation. Moreover a never seen video was used as a test set to evaluate the general capability of the model. Due to the nature of the problem, the Mean Square Error (MSE) was used as a loss function, monitored also in the early stopping callback, in order to prevent overfitting with a patience of 12 epochs. Moreover the Mean Absolute Error (MAE) was monitored as an additional metric on the ReduceLROnPlateau callback, to properly adjust the learning rate during the training phase; in particular, the patience was fixed equal to 6 with a factor of 0.9.

Specifically, the model was trained for 57 epochs, with a learning rate of 0.0001 and a batch size of 12 coupled frames.

Regarding the tranfer-learning model, the one that achived better performance was the ResNet 50. In our study, prior to training, after a first transfer learning where the weights were frozen for the dense layer training, a further fine-tuning was adopted to properly tune the weights of the feature extractor. Meaning that during the training of our model on our dataset, adjustments were made not only to the the fully connected layer but also to the weights of the ResNet.

During training, the model's weights were adjusted to minimize the Mean Squared Error between the predicted values and the actual ground truth values, serving as the loss function. Although the model was initially set to be trained for 50 epochs, the training process utilized early stopping regularization to prevent overfitting. Consequently, the training was terminated within 27 epochs, with the 13th epoch identified as the best-performing epoch. The initial learning rate was set to 0.001. In the event of no improvement within 5 epochs, the learning rate would be reduced by a factor of 0.1, with a minimum learning rate of 1e-15. However, during model training, the minimum learning rate observed was 1e-5.

III. Result

As already mentioned in the Section IIC, the model output describes the camera pose through 3 coordinates for the position and 3 angles for the orientation. In order to visualise graphically the model performance regarding the position, two graphs were plotted for both the scratch and trasnfer-learning model. A never-seen videos was used to assess the general capability of the model. Specifically, Fig 3 shows the comparison between the predicted paths of the 2 models with respect the target one. While Fig 4 show the the point-wise error, the one computed between the corrispondent points in the predicted path and the target one for both the models.

Tab I shows the numerical evalutation of the two adopted models.

Moreover, as mentioned in Section IID, due to the better performance of the scratch model, Fig 5 shows the behaviour of the loss function during the training phase.

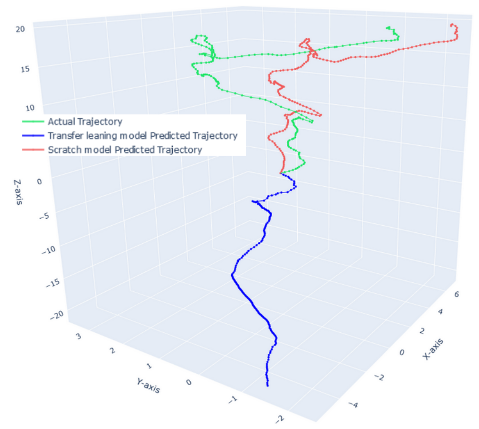

Fig. 3. Comparison between predicted trajectories and target one. Rispectivelly, ground truth trajectory in green, the scratch model prediction in red, and the transfer learning prediction in blue

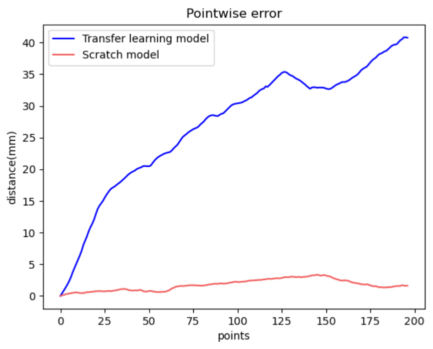

Fig. 4. Point-wise error

	Scratch model	trasf.learn model
Mean Angular Error (degree)	10.7	12.4
Euclidian Distance (mm)	1.7	26.9

TABLE I
NUMERICAL METRICS FOR CAMERA ORIENTATION AND POSITION ESTIMATION PERFORMANCE

IV. Discussion

In Figure 3 is evident that the scratch model provides the most accurate predictions, displaying only a slight offset in the y-axis. Moreover it successfully captures the circular movement of the endoscopic camera but struggles with the fast and subtle movements at the end of the curve. Despite this, it accurately predicts the overall motion in the correct direction of the camera, demonstrating accurate predictions within a certain level of error tolerance.

In contrast, the transfer-learning model captures the general direction of the camera but fails to predict accurately. It comprehends the camera's movement along the Z-axis but predicts in the opposite direction. Although it fails to capture the circular motion of the camera, it accurately identifies where the camera ceases movement. Additionally, the transfer learning model successfully captures the camera's movement along the Y-axis but once again in the opposite direction. Nevertheless, the model successfully captures the overall shape of the trajectory.

Figure 4 illustrates the point-by-point difference between the predicted trajectory and the actual ground truth trajectory. Notably, the scratch model exhibits a constant error throughout the trajectory, represented as a small offset from the actual ground truth trajectory. Conversely, the transfer learning model's prediction shows a linearly increasing error. This behavior is attributed to the model accurately predicting the movement along the correct axis but in the wrong direction. The observed bump in the center of the graph corresponds to a section where the camera executed a circular motion. In this instance, the transfer learning-model failed to capture the circular movement.

Regarding the numerical metrics, the Euclidian distance allows to quantify the model performance in a more objective way. On the other hand, the Mean Angula Error is used in order to assess how good the models are in predicting the camera orientation. As shown in Table I, both the scratch and the tranfer learnign model show a similar and quite robust performance in constrant with the huge differences in the position predictions.

V. Conclusion

This study introduced two models for camera pose prediction: one trained from scratch and the other utilizing transfer learning with models pretrained on extensive datasets. The achieved results serve as a preliminary outcome for a broader investigation. To enhance performance, future improvements could involve training various lightweight models and ensemble them to create a more robust model capable of providing more accurate predictions, leveraging the diverse capabilities of different models.

Additionally, further preprocessing steps could includes running a correlation on two frames, color-coding the differences in each image, and training a model to predict the camera pose based on single frames' dissimilarities. Although such methods may reduce the complexity of the model, they may offer valuable insights.

Consideration can also be given to training models to handle larger sequences of images, including non-subsequent frames. This would enable the model to generalize well in scenarios involving rapid camera movement, capturing both significant and subtle movements.

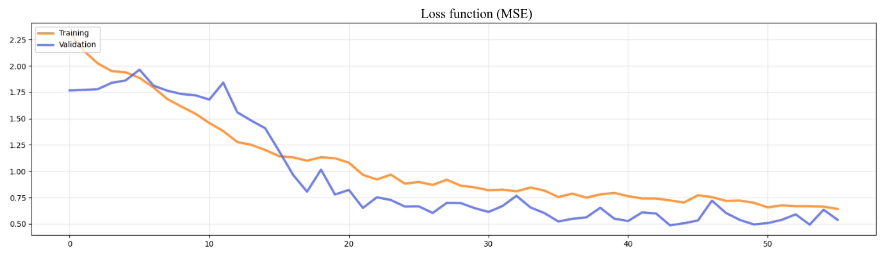

Fig. 5. Trend of the Loss Function

Exploring alternative transfer learning model architectures, particularly those pretrained on endoscopic image sequences, presents another possible improvement for further investigation.

References

[1] Widya, Aji Resindra, et al. ”Whole stomach 3D reconstruction and frame localization from monocular endoscope video.”IEEE Journal of Translational Engineering in Health and Medicine7 (2019): 1-10.

[2] Trinh, Dinh Hoan, et al. ”Mosaicing of images with few textures and strong illumination changes: application to gastroscopic scenes.”2018 25th IEEE International Conference on Image Processing (ICIP). IEEE, 2018.

[3] Behrens, Alexander, et al. ”Local and global panoramic imaging for fluorescence bladder endoscopy.”2009 Annual international conference of the IEEE engineering in medicine and biology society. IEEE, 2009.

[4] Bergen, Tobias, et al. ”A graph-based approach for local and global panorama imaging in cystoscopy.”Medical Imaging 2013: Image-Guided Procedures, Robotic Interventions, and Modeling. Vol. 8671. SPIE, 2013.

[5] Mur-Artal, Raul, and Juan D. Tardós. ”Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras.”IEEE transactions on robotics33.5 (2017): 1255-1262.

[6] O. E. Meslouhi, M. Kardouchi, H. Allali, T. Gadi, and Y. A. Benkaddour, “Automatic detection and inpainting of specularreflections for colposcopic images,” Cent. Eur. J. of Comput. Scien., vol. 1, pp. 341–354, 2011.

[7] Hayoz, Michel, et al. ”Learning how to robustly estimate camera pose in endoscopic videos.”International journal of computer assisted radiology and surgery(2023): 1-8.

[8] Series, B. T. ”Studio encoding parameters of digital television for standard 4: 3 and wide-screen 16: 9 aspect ratios.”International Telecommunication Union, Radiocommunication Sector(2011). 2pt

[9] Hayoz, Michel, et al. ”Learning how to robustly estimate camera pose in endoscopic videos.”International journal of computer assisted radiology and surgery(2023) 2pt

[10] Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Retrieved from http://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/ 2pt

[11] https://opencas.webarchiv.kit.edu/?q=node/30