Bauhaus-Universität Weimar

Final Project

Submission Deadline: 22.09.24, 11:00 pm

Basic Crack Detector

Figure 1. Images showing cracks alongside their respective annotations for semantic segmentation.

Most of the time, when regularly performed, structural health monitoring does an excellent job in ensuring the safe operation of structures of any kind. The detection and annotation of structural defects can be a laborious and time-consuming process. Therefore, the automation of the structure inspection process is a widely researched topic.

Images play a vital role in this process by serving as input for a 3D reconstruction of the structure under inspection and by enabling the automated detection of defects, foremost cracks. (If you are further interested in 3D reconstruction, we recommend attending our course *on Photogrammetrc Computer Visio*n).

State-of-the-art algorithms for crack detection typically rely on artificial neural networks (ANN), which will NOT be subject of the project. In case you are interested in ANNs, please consider attending our advanced course *Deep Learning for Computer Vision*.

In this final project you will use traditional methods to implement a basic crack detector.

Task A – Data Engineering

Data is the fuel for nearly any kind of image-based detection challenge nowadays. If not used for training (i.e. the learning of suitable model parameters), a dataset must be present at least for evaluating the performance of a proposed approach. Thus, the acquisition and preparation of data forms a vital step in designing and developing a detection system.

a) ***Data acquisition***: Take a walk around an urban area and look out for cracks in buildings. (Hint: Please focus on cracks in walls rather than pavement cracks, which can be filled with sand or other materials and can differ in appearance.) Use your camera, e.g. the camera of your phone, to take pictures of cracks. Important: respect the legal restrictions when taking pictures of other peoples' property. For buildings that primarily refers to:
   - Stay on public ground (do not enter private ground without permission).
   - Do not invade the owner's privacy by using aids such as drones, sticks, ladders, etc.
   - Make sure to not capture artwork in a way violating copyright law.
   - Preserve peoples' anonymity by not capturing other people, the doorbell nameplates, etc.

   Document where you captured the image with a precise textual description. Capture at least 10 images.

Computer Vision in Engineering – Prof. Dr. Rodehorst
M.Sc. Mariya Kaisheva
mariya.kaisheva@uni-weimar.de

b) *Data annotation*: Supervised learning requires annotations (also referred to as labels or ground truth) for each input image. For semantic segmentation these labels are of the same height and width as the input image. The pixel values represent the true class of the pixel. It is recommended to use values 0 for no-crack and 255 for crack. Figure 1 shows input images alongside their respective annotations. For annotation you can use basic image manipulation software such as GIMP or more advanced tools such as the Computer Vision Annotation Tool (CVAT).

c) *Data split*: In order to evaluate the performance of your approach, you keep the dataset used during development separate from a test set. Thus, split your dataset into a subset for training/development ($\approx 80\%$) and testing ($\approx 20\%$). The test set should only be used once on the final version of your approach.

d) *Data augmentation*: It can optionally be helpful to use data augmentation on the acquired dataset. By applying basic transformations such as rotation, flipping, contrast shrinking, brightness shift, etc. the dataset can artificially be expanded without need of additional data collection or annotation.

e) *Datasets statistics*: Present basic statistics of the dataset such as the number of images, number of pixels, intensity distribution, the number of labeled pixels, etc. in tables and/or plots.

Task B – Crack Segmentation

In this task you propose an approach to semantically segment the cracks in the image. Semantic segmentation is the task where every pixel in the input image is assigned a class label in the output. Crack segmentation is a binary task with the classes no-crack and crack. As shown in Figure 1, it is recommended to use value 0 for no-crack and 255 for crack.

a) Cracks typically appear as dark line-like structures on brighter background. Thus, ***thresholding*** can be an intuitive first processing step for crack segmentation. Design and implement a thresholding method suitable for segmenting the cracks. Try to incorporate adaptiveness towards images of different brightness levels. The thresholding procedure usually yields many false positives, i.e. regions which do not represent cracks but are classified as such.

b) Play around with ***morphological operators*** in order to clean up the result achieved in the previous step (e.g. by removing isolated points and ensuring a better connectedness of the detected cracks). You are allowed to use built-in functions.

c) From the resulting segmentation mask extract discrete regions by implementing ***connected component*** analysis as presented in the lecture.

d) *Feature engineering*: Create a feature vector for each discrete region instance of potentially useful features (region properties such as lenght, circularity, color, ...) – be creative. Based on the annotated ground truth, assign a label to each region whether it represents a crack or no-crack. These feature vectors alongside the true labels will serve as input to the next step.

e) Classifier: Implement a classifier that uses the prepared features in order to filter out non-crack regions. You are allowed to implement a heuristic rule-based approach. Furthermore, you are welcome to experiment with support vector machines (SVM), decision trees, or any other concept

taught in the course, which you consider suitable. Explain your reasons and ideas for choosing your approach.

Task C – Crack Analytics

Crack segmentation unfolds its power only when the results undergo further processing. Information such as the length or the number of branches of a crack are crucial for the assessment of a structure's condition.

a) Implement a ***metric*** to assess the performance of the implemented approach. The standard metric for semantic segmentation is intersection-over-union (IoU). Report the performance of the detector on the test set and discuss the adequacy of the metric for crack detection.

b) For ***further processing*** use thinning in order to reduce the segmentation results to a line-like representation of the crack. You are allowed to use built-in functions.

c) Implement a function ***to compute the length*** of the detected cracks in the test set. Report the lengths.

d) Comment on the ***usefulness of the implemented approach*** for practical crack detection. What are the strengths and weaknesses of your approach?

Sample Results: