Detecting Healthy Concrete Surfaces

Philipp Hüthwohl[1], Ioannis Brilakis[2]

[1]Ph.D. Candidate, Dept. of Engineering, Univ. of Cambridge, Trumpington St., Cambridge CB2 1PZ, U.K. (corresponding author). E-mail: ph463@cam.ac.uk (corresponding author)

[2]Laing O'Rourke Reader, Dept. of Engineering, Univ. of Cambridge, Trumpington St., Cambridge CB2 1PZ, U.K. E-mail: ib340@cam.ac.uk

Abstract

Teams of engineers visually inspect more than half a million bridges per year in the US and EU. There is clear evidence to suggest that they are not able to meet all bridge inspection guideline requirements due to a combination of the level of detail expected, the limited time available and the large area of bridge surfaces to be inspected. Methods have been proposed to address this problem through damage detection in visual data, yet the inspection load remains high. This paper proposes a method to tackle this problem by detecting (and disregarding) healthy concrete areas that comprise over 80-90% of the total area. The originality of this work lies in the method's slicing and merging to enable the sequential processing of high resolution bridge surface textures with a state of the art classifier to distinguish between healthy and potentially unhealthy surface texture. Morphological operators are then used to generate an outline mask to highlight the classification results in the surface texture. The training and validation set consists of 1,028 images taken from multiple Department of Transportation bridge inspection databases and data collection from ten highway bridges around Cambridge. The presented method achieves a search space reduction for an inspector of 90.1% with a risk of missing a defect patch of 8.2%. This work is of great significance for bridge inspectors as they are now able to spend more time on assessing potentially unhealthy surface regions instead of searching for these needles in a mainly healthy concrete surface haystack.

Keywords: Bridge inspection; Defect detection; Automated bridge inspection; Healthy concrete;

1. Introduction

Bridges are the most critical and complex structures in a road network, both technically and strategically. Weight-limitations or closures have negative consequences on the economic success of a country as well as on the user satisfaction. Bridge inspections need to be carried out to know the bridge condition, to collect information about damages and to make appropriate operational or maintenance decisions (load limitations, maintenance needs or closure). A team of engineers inspect a bridge manually on site regularly (typically every two years a general, purely visual inspection, every five years a more detailed in-depth inspection from a touching distance including the use of tools) (The Highways Agency 2007).

Bridge inspection guidelines require engineers to visually identify both small and large defects (e.g. down to 0.3 mm in width for cracks) on all bridge element surfaces (The Highways Agency 2007). Our datasets show that the concrete surface area of an average highway bridge taken around Cambridge is 2,440 square meters, equal the size of almost six basketball courts. Inspecting this takes more than 20 hours if allowing for 30 seconds inspection time per square meter to identify potentially unhealthy areas, closely examine these areas, taking measurements and documenting the defects in writing and visually. Moreover, this is the pure inspection time, not accounting for time required to perform safety measures, walking and climbing to get into a solid inspection position or rest periods. In addition, there usually exists a serious issue of accessibility where some areas to be inspected are not easily accessible. Image timestamps of 399 inspections were analysed to learn about inspection duration. The time span between the first and last image allows a conclusion on the duration of the visual part of an inspection based on the assumption that an inspector regularly takes images during a visual inspection. The average time for a general inspection was 19 minutes and for an in-depth inspection 72 minutes. It is therefore questionable whether an inspector is able to inspect the entire bridge surface with the required level-of-detail and from a distance from which all defect types can be identified. Inspectors have to make a trade-off between inspection time and inspection distance. They do it in two ways: (1) Inspectors might look from a distance where they are unable to see small details or (2) Inspectors might skip some surface parts because it is too time-consuming to get into a position from where the surface is visible. As a result, bridge condition information is incomplete.

Missing out small details leads to missing minor defects. Preventive maintenance, which means to maintain minor defects before they become major, can reduce costs by up to 65% (Kong et al. 2003). More importantly, skipping surface area completely bears the risk of missing major defects which can lead to major closings or a complete loss of structural integrity and fatal accidents (Xie and Levinson 2011).

1.1. State of practice

Inspectors perform two tasks during an inspection: First, they identify which areas of a bridge are prone to and critical for defects. This is done empirically and based on a subjective structural interpretation; no defined rules exist. Typical areas are the ones close to a support (e.g. the connection between column and girder) or with maximal bending (e.g. middle of span). Second, an inspector looks for potentially unhealthy spots in the critical areas. Only these potentially unhealthy spots are examined more closely by conducting four steps: take a close look to identify the defect type and possible cause; take measurements of the relevant defect properties; give a condition rating based on the measurements and the inspection guidelines and finally document findings in writing and figurative in a sketch or an image (Spencer 1996). This second step typically affects only a minor surface area; most of the surface is non-deficient concrete.

1.2. State of research

Technologies for collecting the as-is raw data of a bridge exist: Laser scanning or Structure from Motion (SfM) can provide high-precision dense point cloud data with registered imagery. Methods for manual or automated as-is modelling exist (Hüthwohl et al. 2016; Lu and Brilakis 2017). High resolution imagery can be used for texturing elements. Textures are stored in common 2D image formats such as jpeg or png. UV mapping is the process of applying a flat, two-dimensional image onto a three-dimensional shaped object (Murdock 2008). With these methods, a fully textured as-is digital representation from a real structure can be compiled such as the one shown in Figure 1. The textures include very small surface details from the real surface such as cracks, aggregate and spider nets. Hence, these models can be used to research, if they are sufficient for manual or automated defect detection.

Figure 1: 3D as-is model of fully-textured RC highway bridge, deck view on the left, bottom view on the right

Any method, automated or manual, has to achieve at least the same inspection quality as the state of practice: a team of human inspectors on site. Two metrics define the level of inspection quality for the scope of this work: the risk of missing a defect and the ability to generalize over healthy and potentially unhealthy areas.

Determining the performance of existing inspection schemes regarding the risk of missing a defect is difficult. No up-to-date study exists. Phares et al. (2004) did an investigative study to evaluate the performance of bridge inspectors. They found out that inspectors tend to miss documenting 46% of the defects. Authorities adopted inspection schemes since then. One of the adoptions was to change from a component inspection level to an element inspection level. A performance evaluation of the new scheme is missing. Hence, any automated inspection method has to have a considerably lower risk of missing a defect than the one determined by Phares et al. The second metric, generalization, is difficult to measure quantitatively for the scope of this work. Nevertheless, it is an absolute requirement for the evaluation. Human inspectors generalize well, as they are able to identify and examine suspicious areas based on their experience even if inspection guidelines do not list rare, untypical types of defects.

1.2.1. Appearance of healthy and potentially unhealthy concrete

A general definition of the appearance of potentially unhealthy or healthy concrete does not exist. Newly build reinforced concrete is approximately homogeneously coloured. The admixed aggregate and sand appear as small spots in different colours (depending on the mixture, white, shades of brown, almost black).

Multiple influences immediately change the appearance of a concrete surface already during construction. For example, shrinkage during hardening and design loads plus gravity, traffic and environmental loads lead to initial capillary cracks in the concrete. These cracks are difficult to see with the naked eye and do not constitute a defect, hence are not to be considered as potentially unhealthy. Formwork marks, minor corroding metal pieces (e.g. nails left from construction) and differences in concrete texture are also common and occur frequently on concrete surfaces. Environmental influences such as rain, vegetation or dirt change the concrete surface texture over time. These influences vary depending on the location and exposure. Momentary environmental conditions during the data collection, such as strong sun or rain, have an additional effect on the image texture. Figure 2 shows multiple examples of such normal patterns: (a) dust and spider webs, (b) formwork marks, (c) water stains and (d) strong shadows.

Potentially unhealthy areas are all areas relevant for an inspector to take a close look for the scope of this work. These are primarily concrete defects, but also include signs of vandalism, graffiti and littering. Inspection manuals list typical examples of concrete defects. Huethwohl et al. (2017) analysed multiple inspection manuals from different continents. Spalls (e), cracks (f), rust stains (g), efflorescence (h), scaling and abrasion / wear are the most common ones and pictured in Figure 2.

Methods for detecting potentially unhealthy / healthy concrete use a two-dimensional image as input. The three-dimensional shape is irrelevant for most considered defect classes, as long as the texture image is undistorted. Abrasion / wear is the only defect class that primarily affects the shape. Abrasion / wear is excluded for the scope of this work as these defects are not visually detectable in 2D images and state of the art as-is models do not model such minor shape deformations.

Figure 2: Concrete texture examples: (a) dust, dirt and spider webs, (b) formwork marks, (c) water stains, (d) strong shadows, (e) spall, (f) crack, (g) rust stain, (h) efflorescence

1.2.2. Research on concrete defects

The research community has shown interest in tackling the challenge of separating potentially unhealthy from healthy concrete, yet has not been able to entirely address the problem for bridges. General approaches directly address the problem of distinguishing potentially unhealthy and healthy areas in one step by using a single metric for all possible potentially unhealthy candidates. McRobbie et al. (2007) tested fifteen different feature descriptors such as entropy, standard deviation, mean value of area, quadtree decomposition and different edge detectors on a deteriorated bridge abutment wall, where the surface texture was reconstructed from multiple images. They used a grid-based feature threshold for the classification based on the assumption that large variations in the feature descriptor imply potentially unhealthy areas. Yet, none of the metrics was able to reliably distinguish between the two classes.

A different approach is to combine multiple single defect class detectors to a combined multi-classifier. A considerable number of single class detectors exist, mostly for detecting cracks. These detectors, however, address a different problem and are out of scope of this work. Koch et al. (2015) have recently done a thorough review on this. They concluded that crack detector methods need improvement. They are prone to noisy data, changing lighting conditions, and still require a significant amount of user input. McRobbie et al. (2011) examined in addition to the afore mentioned fifteen separate detection metrics, if potentially unhealthy concrete areas can be detected by combining multiple sufficiently uncorrelated metrics. They found out, that this approach could slightly improve the detection results. Quantitative results are not given. There are two limitations why these combined methods cannot reliably solve the problem of identifying potentially unhealthy areas on bridge element texture. First, single class detectors detect only one single defect type by design. A combination of single class detectors is only able to detect the defects out of the combined single class detectors. Inspectors, however, have to detect potential defects even if they are not listed in the defect catalogue. Secondly, single class detectors use a dataset containing mostly samples of their specific target class for validation; their performance with images from other defect classes in unclear.

Change detection is the process of identifying differences in a set of multi-temporal datasets, primarily images. This field is well-studied in different disciplines such as remote sensing (Lu et al. 2004), surveillance (Collins et al. 2000) or medical diagnoses (Bosc et al. 2003). Multiple researchers have exploited this technique for civil infrastructure, mostly tunnels. The key idea is that an initial dataset represents a faultless structure. Changes to a second dataset, taken at a later point in time, automatically present a potential defect and hence are potentially unhealthy. The difficulty, however, is to find a difference metric that is robust to changes in lighting and imperfect registration. Guo et al. (2009) tested the image difference after an image registration and pre-processing step on images from a storm-water pipe segment. Based on the intensity difference, a threshold determines for each pixel if it has changed or not, hence if it is potentially unhealthy or healthy. The per-defect accuracy was 84% with a false positive rate (FPR) of 21%. Stent et al. (2015) trained a convolutional neural network (CNN) to detect change in tunnel linings. Two registered images for comparison were loaded into separate input channels of the CNN. The training dataset comprised of real tunnel surface texture with artificially added defects. 84% of the changes were detected with a per pixel FPR of 10%. Change detection has three major disadvantages for the scope of automated bridge inspection: First, images need to be perfectly aligned down to a pixel level. This is already difficult for simple geometries such as tunnels, but even more so in the case of more complex structures such as bridges. Secondly, change detection is prone to major changes in lighting. Lighting can be controlled up to a certain degree indoors or inside a closed structure. Certainly it can not be controlled at a bridge with reasonable effort. Thirdly, one might miss a defect completely if relying on changes as the sole indication for potentially unhealthy areas. This happens if in one case an inspector misses a defect and marks an area as healthy. If it does not change until the next inspection cycle, it will still be considered healthy as it was already present and labeled as healthy during the last inspection.

Additional sensors have been investigated regarding their possible use for bridge inspections. This study focuses on the data analysis rather than presenting technical details on the sensors. Matsumoto et al. (2012) used a thermal camera to identify internal defects on a bridge deck. A heatmap is automatically generated based on the local temperature profile. Potentially unhealthy areas are highlighted based on a simple temperature gradient. Quantitative results are not given. Lerma et al. (Lerma et al. 2011) utilized a thermographic camera to detect moisture. Both studies are able to detect invisible sub-surface defects. But other crucial interest classes such as cracks are not detectable. Jahanshahi et al. (2013) used a depth sensor (Microsoft Kinect) to automatically detect pavement defects. Their approach is based on detecting major deviations from a fitted plane within the three-dimensional depth data. As before, shape-based methods lack in completeness regarding different defect classes which do not involve significant change on the 3D shape. Valença et al. (2017) combined a digital camera and a laser scanner in order to utilise the precision of laser scanning with the high resolution of image processing. Reference points are used for orthorectifying the image which is then used for crack property extraction.

Conservative image classification approaches typically use handcrafted features (edges, corners, gradient variations, etc.) and a simple classifier (threshold) to make a classification decision. This is based on the assumption that one understands how to detect specific things, such as cracks, based on assumptions and simplifications such as changes in contrast or colour. In fact, it is unknown how humans identify defects on a surface. Hand-crafted features are our best guess. Increase of performance of hand-crafted feature methods have stagnated in recent years (Jia et al. 2014)

1.2.3. State-of-the-art classification / segmentation approach

In contrast, deep learning has outperformed humans in classification tasks such as recognizing handwritten digits (Cireşan et al. 2012) and skin cancer classification (Esteva et al. 2017). An artificial neural network with multiple hidden layers is trained using an extensive, typically labelled dataset. The training is end to end, meaning that raw image pixel values form the input of the network and the output directly presents the desired output format. Training algorithms use gradient descent to converge towards a local minimum. Hidden layers form a hierarchical feature set and each layer models more complex data based on the predecessors. This way, relevant features for a specific classification task are learned automatically without the need of hand-crafted features. The Inception

Resnet network has outperformed previous state of the art models for image classification on the common academic image classification dataset ImageNet achieving 80.4% top-1 accuracy and 95.3% top-5 accuracy (Szegedy et al. 2016). The ImageNet dataset contains a wide range of categories such as animals, flowers, sports, vehicles and persons. Deep learning, however, is a relatively simple brute force approach, it requires an extensive dataset and it results in a black box. If and how it will converge and how it will come to a classification decision is unclear and a special focus of current computer science research. A rule of thumb is that the size of the training dataset and the number of variables in the network should be equal in size. The mentioned inception network roughly has 50 million parameters (Alex Alemi 2016).

Fully convolutional deep neural networks (CNN) for semantic segmentation are networks that directly output a semantic map of an input image (Long et al. 2014). Each image pixel has a class assignment; clusters of neighbouring pixels represent a class instance. This has been popular for autonomous driving to locate asphalt, road signs, trees, pedestrians, etc. (Badrinarayanan et al. 2015). Semantic texton forests (STFs) can be used for semantic segmentation as done by Golparvar-Fard et al. (2015) for road scene segmentation or Radopoulou and Brilakis (2016) for road surface inspection, yet have been outperformed by CNNs regarding their classification accuracy. The input size to CNNs for images is, even with enormous computing power, still very limited. A typical input size is 512x512 pixels or 0.26 megapixel (MP). A resolution of 0.1 mm^2 per Pixel is needed for reliably detecting 0.3 mm wide cracks (Marks 1991). Following this, an element surface of only 1 m^2 requires a resolution of 100 MP, more than 380 times the size of what is possible to be processed by a CNN today. In addition, a representative and labelled dataset sufficient in size for training a neural network to detect potentially unhealthy areas on concrete bridge elements does not exist.

1.3. Gaps in Knowledge and Research Questions

Hence, the research body does not present a method to reliably reduce the visual search space for a bridge inspection to potentially unhealthy regions of interest only. A new approach is needed that can separate potentially unhealthy from healthy concrete based on surface texture images to drastically increase the inspection efficiency. This has to happen on a real life dataset with limited FPR. Existing work focuses on detecting a specific type of defect, works only for simple geometrical structures and limited environmental conditions or is technically not able to deal with the massive image resolution given by inspection requirements and is therefore not able to solve the problem. Deep learning, instead, has achieved promising results. Yet it has not been researched entirely on how to fully utilise it for the scope of bridge inspection.

This work's objective is to present a method that is capable of separating potentially unhealthy from healthy concrete by automatically assigning a class label (potentially unhealthy / healthy) to each location (patch or pixel). The significance of detecting healthy concrete derives from the fact that there are no effective defect detection methods findable in the body of knowledge. Defect methods could be more effective if a method is able to separate healthy concrete. In addition, disregarding healthy concrete can save time for both manual inspectors, as well as for automated defect detection. This results in the following research questions: (1) how can high resolution surface texture be split, such that it maintains all necessary details for defect detection but at the same time can be sequentially processed by a state-of-the-art image classifier; (2) which state-of-the-art classifier is suitable; (3) how can the results be merged to represent a meaningful representation to guide inspectors to areas of potentially unhealthy concrete; (4) by how much can this approach reduce the search space for an inspector; (5) what are the risks of this approach to miss a defect?

The following chapter "Proposed solution" details the proposed method of splitting, classifying and merging the results. "Research Methodology and Results" presents the dataset, data sources, the label assignments and the experiments carried out in order to prove the performance. Finally, "Conclusion" summarizes and discusses the results, benefits and limitations of this presented work along with an interpretation and implication for both, practice and society.

2. Proposed Solution

The proposed method automatically detects potentially unhealthy areas in bridge surface textures. A sliding window approach splits the surface texture into image patches. Then, a pre-trained Inception-v3 network is used and fine-tuned on a domain-specific dataset to classify each image patch separately. The final step merges the classification results to a mask for indicating the different patch labels. Figure 3 depicts a flowchart of our method.

Raw bridge inspection images contain a variety of unrelated image contents with irrelevant parts, such as sky, vegetation, different elements at different scale and perspectives. An increased complexity of

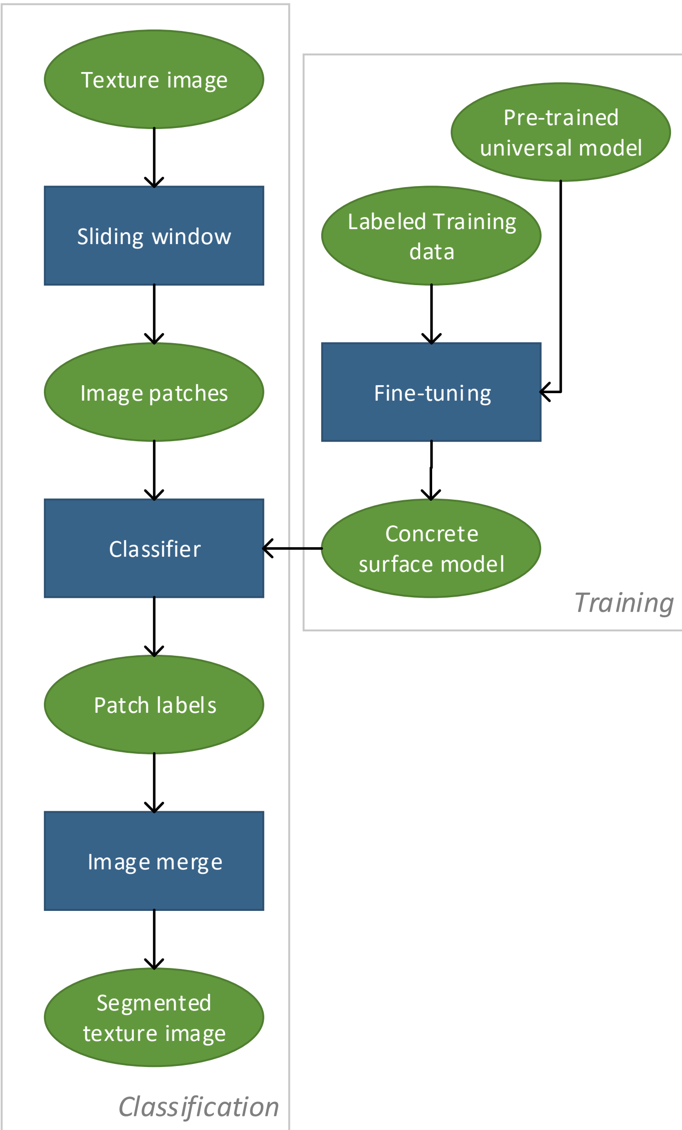

Figure 3: Proposed training and classification method for detecting potentially unhealthy and healthy concrete areas

image contents worsens training and classification quality. In addition, a finding in an unregistered image cannot be easily located on the 3D geometry, which is necessary for a subsequent assessment. For these reasons, a reconstructed surface texture, as in Figure 4, is used as input image. The following assumptions can be made by using the reconstructed surface texture: (1) there is only relevant image content which is either potentially unhealthy or healthy; (2) background is uniformly coloured (black or white); (3) there are no or minor optical distortions; (4) each point of the surface texture represents the corresponding and nearly orthogonal view of that point; there are no or minor perspective distortions (as they have been compensated during surface reconstruction); (5) the surface area mapped by each pixel is constant throughout the image; there are no scale or resolution differences. Minor seams will occur, as the reconstruction process is not able to perfectly stitch the images and completely remove any radial distortions. However, these effects are assumed to be insignificant since both effects can be reduced by an improvement in image acquisition and, moreover, slight seams and distortion does not alter the overall appearance of a defect.

A state-of-the-art classification method shall do the classification task. The GoogleNet Inception v3 architecture, which is a convolutional neural network (CNN), is the second most accurate model on the ILSVRC 2012 image classification benchmark, a common academic image recognition dataset. Only the much deeper Inception ResNet-v2 model has achieved a slightly higher accuracy (+2.4%) but at the price of double the memory and double the computation costs (Alex Alemi 2016).

The input of this classifier is not able to take images of arbitrary size. They are limited to 299x299 pixels. A common approach would be to simply resize the surface texture. This works well for images where classification objects fill a considerable part of an image. The scenario is different for the scope of bridges. Cracks, for example, fill a very small portion of an image. A one meter long and one millimetre wide crack only fills a thousandth of a rather small, one square meter surface. Downscaling this image to a typical network input size would remove this defect completely. For this reason, this work proposes to use a sliding window approach. It converts the surface texture into a size applicable

Figure 4: Reconstructed high-resolution surface texture of a bridge column, uncoiled 2D view on the left, 3D view on the right

as input for a classifier by retaining the original resolution and aspect ratio. Retaining the resolution is crucial particularly for bridge inspection. The sliding window is a fixed-size window that partially copies the source image I into a new image P, which represents one patch. It then slides by a defined offset o before it extracts the next patch. It iterates over both dimensions of the image and terminates as soon as the method has finished extracting patches from the entire input image. The image is extended over the edge in a mirrored manner to handle the edge area identically. Formula 1 theoretically describes the patch extraction. k is the patch number, o the offset between two patches and nx the number of patches in the direction of the first dimension.

$$P(i, j, k) = I\big((k \bmod nx) \cdot o + i, \lfloor k/nx \rfloor \cdot o + j\big) \tag{1}$$

The extracted image patches serve as input for the second step which is the classifier. This step is a binary classifier; the corresponding class labels are potentially unhealthy and healthy. Using semantic segmentation would result in a pixel-based classification. An inspector needs to see the classification result based on a potentially unhealthy area, not on a pixel. Having a pixel-based classification result would therefore require consolidating the classification results to a useful format later on. A patch-based classification decision, instead, is superior as it simplifies this consolidation step and results can be directly presented to an inspector.

The final step post-processes patch labels into a mask which outlines potentially unhealthy areas as areas of interest for an inspector. Patch regions are partly overlapping which can result in having different class labels for the same location in an image. Image locations with at least one potentially unhealthy label is marked as such. This is a conservative approach; missing out potentially unhealthy areas is crucial to the overall validation, whereas having a healthy area labelled as potentially unhealthy is less critical. A binary mask in a grayscale image represents the class labels. This process is demonstrated in Figure 5. A value of 0 represents the label healthy, 255 represents a potentially

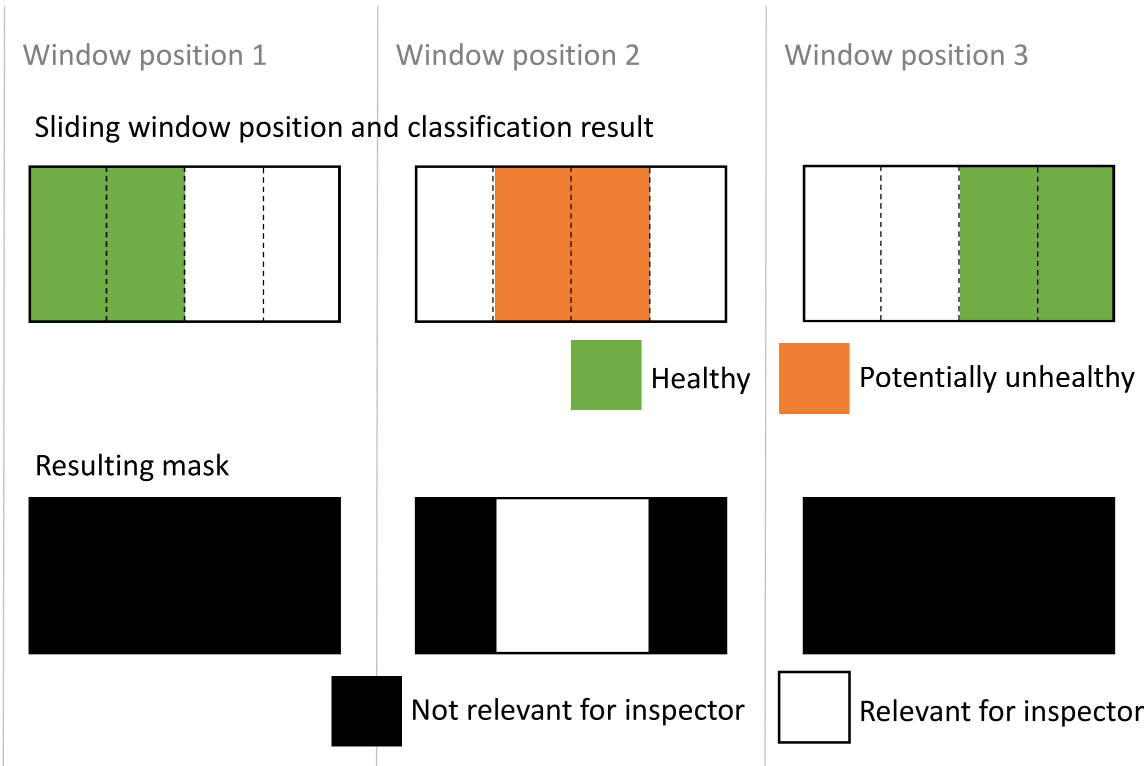

Figure 5: Post-processing overlapping classification results to build masks of potentially unhealthy concrete.

unhealthy label. Potentially unhealthy areas in the texture files are framed to retain a maximum of the original surface texture. Binary morphology reduces the mask to the outline only as described in Formula 2 where O is the binary image showing the outlines; M is the binary mask and H the structuring element. First, the mask M is eroded using a standard 4-neighbour structuring element H which is then inverted. Multiple eroding iterations determine the border width. Secondly, the intersection of the resulting inverted and eroded mask with the original mask M leads to the outline mask O which is set to red in the output image.

$$O = M \cap \overline{M \ominus H} \qquad (2)$$

This study hypothesizes, that our presented slicing and merging algorithm in combination with a state-of-the-art image classifier can outperform existing healthy concrete detectors. It is tested on a manually labelled dataset using the two performance metrics: The first one is the reduction of search space for an inspector in order to answer the question of how much of the surface texture can be skipped by an inspector without increasing the risk of missing a defect. This is equivalent to true negative rate (TNR) stated in Formula 3, where TN is the healthy concrete area that is correctly classified as such divided by the area that was correctly classified as negative (TN) plus the area that was falsely classified as positive (FP).

$$TNR = TN/(TN + FP) \qquad (3)$$

The second metric is the likelihood of missing a defect in order to find the risk of actually missing a defect. This is the false negative rate (FNR) which is calculated as stated in Formula 4. The falsely as negative classified area (FN) divided by the sum of correctly positive classified area (TP) and the area which is falsely classified as negative (FN).

$$FNR = FN/(TP + FN) \qquad (4)$$

3. Results and Discussion

The assumption is that inspectors could spend more time per defect if they were able to focus on potentially unhealthy areas only. In general, it is the idea that complexity of the problem can be substantially alleviated if an inspector is guided in the decision where to look for defects. Inverting the problem by identifying areas that are obviously without problems does generally take much less effort than identifying problematic areas by taking a closer look.

3.1. Dataset preparation

Classification performance directly depends on the quality of a corresponding training dataset. Hence, its composition requires special care. A labelled dataset for the scope of this work is not publically available. Cambridge Bridge Inspection Dataset (Huethwohl 2017) is a newly composed dataset which is part of this work and is based on two data sources: The first one is from our own data collection. 21,284 high resolution images (42 MP) from 10 RC highway bridges around Cambridge were collected for the scope of this work, out of which 17,124 images cover the main parts of the bridges (deck, columns, piers and abutment; 4,160 images from non-concrete side walls and basements were excluded). These images, however, do not contain a sufficient number and variety of defects as the bridges are in a good condition. Departments of Transportation (DoT) or their contractors maintain bridge management systems (BMS) which contain inspection and condition information, in particular defect images taken during inspections. Atkins, the U.S. Federal Highway Administration (FHWA) and the Georgia DoT have kindly granted access to 22,121 of their inspection images, which sums to 39,245 raw image candidates. Still, image labels are missing for utilizing the image candidates for classifier training. Hence, images need to be manually labelled. Image content consists of three label types, out of which two are relevant to this work: Healthy areas (concrete in various colours and appearances), potentially unhealthy areas (defects, discolorations, etc.) plus background noise (sky, asphalt, vegetation, workers, cars, etc.). As many of the images are just single, random images from bridges, there is no way to use them for surface texture reconstruction. Nevertheless, they can be used for training and validation based only on the relevant parts of the images. The naïve approach is to label each image separately and assigning a label on a pixel level. This labour-intensive task, however, would be unreasonably time consuming to achieve. Picking only a subset of images instead for manual labelling risks compiling a biased training set with only easy samples. To overcome this, Figure 6 illustrates a newly established process for randomly extracting and manually labelling image patches. It starts with randomly selecting an image (uniformly distributed) from all candidates and then extracting a squared patch with a randomly selected window length (normal distributed, mean at window size and variance of a tenth of the window size) and at a randomly selected position (uniformly distributed). The classifier is trained to be independent of the size of the image area covered on the surface (surface resolution), although the input size of the network is invariable (with a pixel size of 299x299). This is achieved by the fact that the patches of the training data cover a different surface area size. The patch is resized to the input size of the network and then manually assigned with a patch-based label following two decisions: The first decision is if the patch only contains concrete. If this is not the case, the patch is discarded and a new patch is drawn. If the patch, instead, contains concrete only, the second decision follows: to manually assign the relevant labels healthy and potentially unhealthy. Only the patch area is considered; surroundings of the image patch are not included for the labelling decision. At this point, one could criticize that a subjective labelling decision directly influences the classifier outcome. The key point is to conservatively assign the label. If in any doubt that a patch shows an area of concern, it gets the potentially unhealthy label. In addition, the labelling and the training is a repeatable process which can be analysed and optimized in more detail for the close decision patches. The dataset has 1,028 labelled and randomly picked patches out of which 896 build the training subset and 132 build the evaluation subset. Table breaks down the number of images in regards to class labels, data sources and training / evaluation dataset.

Table 1: Break down of image patch numbers with respect to label and data source

	Cambridge Data Collection	DoT Inspection Data	Total
Potentially unhealthy	Train: 70 Eval: 11 Total: 81	Train: 206 Eval: 50 Total: 256	Train: 276 Eval: 61 Total: 337
Healthy	Train: 473 Eval: 54 Total: 527	Train: 147 Eval: 17 Total: 164	Train: 620 Eval: 71 Total: 691
Total	Train: 543 Eval: 65 Total: 608	Train: 353 Eval: 67 Total: 420	Train: 896 Eval: 132 Total: 1028

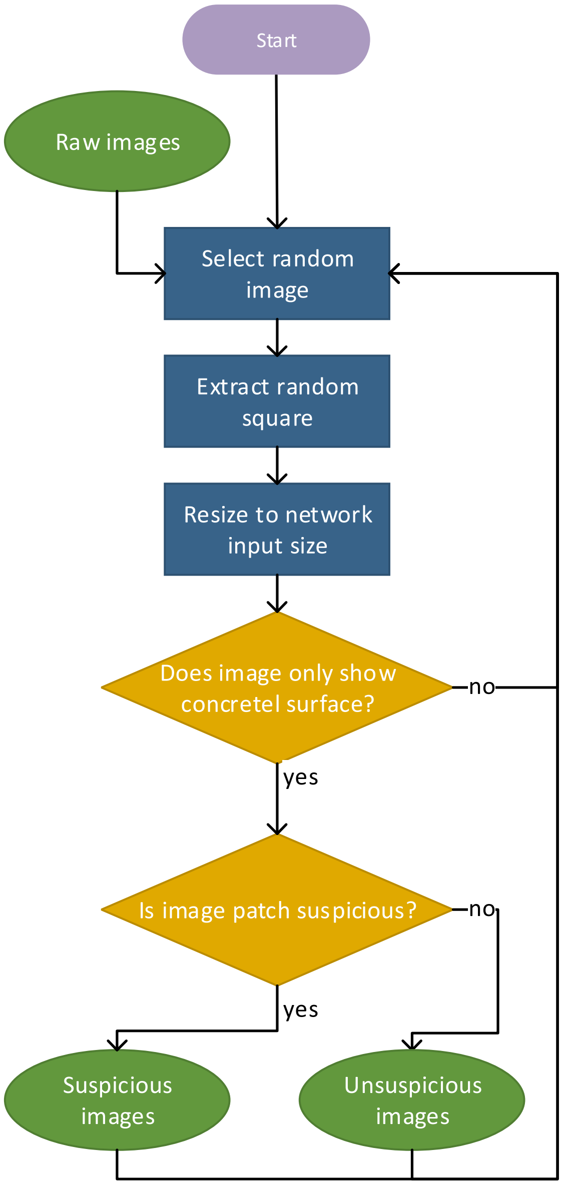

Figure 6: Process to extract random patches and to manually assign image labels

The input layer of the neural network determines the input size in pixels of the image. Resizing the pixel input size of a trained network is not possible without losing the trained weights. The pre-trained network has an input size of 299x299 pixels, and hence, all patches in our dataset have a pixel size of 299x299 pixels.

3.2. Implementation and network training

Gygax is a research platform developed at Cambridge that allows researchers to simultaneously access BIM, image, video, and/or point cloud data, as well as to load them in memory, visualize them and process them simultaneously (Huethwohl et al. 2017). The platform already supports textured as-is bridge models. Google's open-source software library Tensorflow 1.2.1 provides strong machine learning functionality (Abadi et al. 2016). Tensorflow was integrated into Gygax by using TensorFlowSharp, which wraps the Tensorflow C API as a strongly-typed .NET API for the use from C# (Icaza et al. 2017). An implementation of the inception network along with a pre-trained model exists. Gradient descent Root Mean Square Propagation (RMSProp) trained the network for 300,000 steps or about 330 epochs after reducing and randomly initializing the output layer with a batch size of 32, an initial learning rate of 0.001 and a learning rate decay factor of 0.16. A change of the hyper-parameters was not tested as the moving average over the loss-function converged. Different hyper-parameter sets were not tested as these do have minor impact on the training quality for the scope of this work. They rather control the stability and speed of convergence.

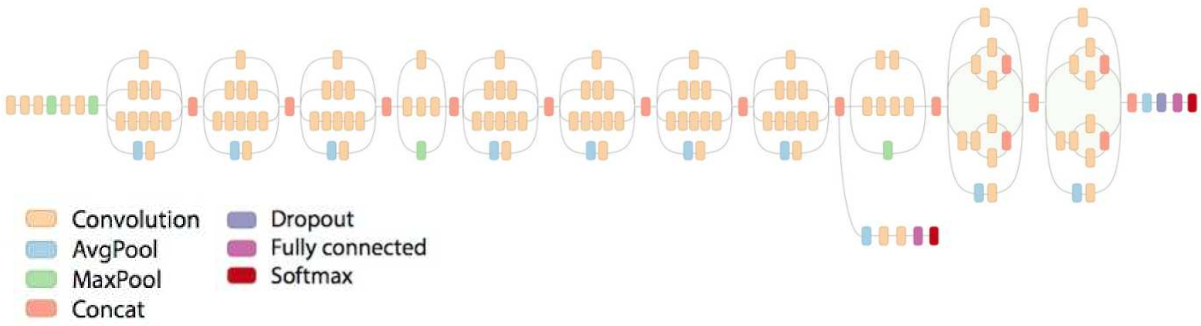

Figure 7: GoogleNet Inception v3 network with the nodes highlighted for adjustment (Alex Alemi 2016)

Figure 7 shows the working system of the deep neural network. The network consists of multiple stacked inception modules. Each inception module consists of three or four parallel convolution operations with varying filter sizes and one max or average pooling step. The network roughly needs five billion multiply-adds per inference and has less than 25 million parameters. Our bridge inspection dataset is too small to train such a network from scratch for the stated purpose. Splitting the training into two phases based on transfer learning overcomes this; the key idea is that meaningful features in one feature space can be transferred to a different one. The first phase is a general training. An on 1,000 classes and 1.2 million images pre-trained ImageNet model is used. This pre-trained network is publically available. The second stage is the fine-tuning. The number of labels is changed in the final classification layer to two and use the weights from the pre-trained model except for the modified final layer. This final layer is assigned with random weights and then fine-tune the network using our own dataset. Two alternative training strategies exist: Training a full network means to input one training sample into the network, then calculate the outcome based on current weights and back-propagate the difference to the desired label. The back-propagation changes the weights based on the participation in the decision-making and the hyper-parameters (such as learning rate). This update happens through all layers of the network. It is computationally expensive and leads to the best possible accuracy. A less computationally expensive approach is to only update the newly initialized weights and to back propagate the training samples only through the last layer of the network. This strategy assumes that a new classification task can utilize relatively high abstracted features from some of the pre-trained model classes to describe new classes. In simple terms, this assumes that new classes are visually close to one of the pre-trained classes. The presented work follows the complete training and back-propagation through the whole network, as the goal is to find a reliable classification result rather than reducing computational costs. The training results in a concrete surface model that is able to label each image patch separately as potentially unhealthy or healthy concrete.

Both, training and evaluation ran on a dual GPU system with two GeForce GTX 1070 and 8 GB of GPU memory each, an Intel Core i7 4 GHz CPU and 32 GB system memory. Execution of 300,000 steps took 67 hours on this machine.

3.3. Experiments

Two experiments evaluated the performance of the presented method. First, stability of the classification result was determined by classifying the evaluation dataset only using the trained network and a bias. The outcome of the classification can be interpreted as a likelihood of the patch belonging to either the potentially unhealthy or the healthy class. This assignment is typically done based on the maximal class score. A tendency to classify a patch as potentially unhealthy in case of doubt is appropriate in order to minimizing the number of missed defects. The potentially unhealthy class is preferably selected if class scores are close. More precisely, the difference between the two classes is used as a control variable to understand the stability and balance between false negatives, which come at very high costs, and false positives which are annoying but tolerated. Figure 8 presents the results. The horizontal axis represents the difference threshold to determine the class assignment. The vertical axis represents the value of four different measures:

Precision, as defined in Formula 5, is the fraction of samples classified as positive and actually being a positive sample. This measure alone is not sufficient as a classifier that classifies everything as negative (except one sample to avoid division by 0) would result in a high *precision* value and hence, would misleadingly be assessed as positive.

$$Precision = TP/(TP + FP) \quad (5)$$

Therefore, *recall* was introduced which also considers the false negative classified samples. This in turn would lead to good results if all samples are classified as positive. It is the fraction of positive samples that actually were classified as positive as in Formula 6.

$$Recall = TP/(TP + FN) \tag{6}$$

Accuracy is a combined measure that takes TP, TN, FP, and FN into account. However, it depends on a balanced number of positive and negative samples.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{7}$$

The F_1 score in Formula 8 was introduced to overcome all stated limitations. It is the harmonic mean between precision and recall.

$$F_1\ Score = 2 \cdot Precision \cdot Recall/(Precision + Recall) \tag{8}$$

The F_1 score in Figure 8 shows a steep rise and fall in the range of -1 to -0.8 and 0.8 to 1.0. In between, there is a stable plateau of around 0.8 with a slight increase and a peak of about 0.92 at a threshold value of 0.5 towards the tendency to label patches as potentially unhealthy. The maximum F_1 score of 0.90 was achieved at a bias of 0.61. This illustrates that the classifier is able to reliably distinguish between potentially unhealthy and healthy patches and is able to do this in a very stable and robust way (relatively independent from the threshold). Figure 9 gives examples of the classification results for a difference threshold of 0.5 for the group of true positives, true negatives, false positives and false negatives. The classifier is able to distinguish between the two classes over a variety of different defect types and concrete appearances. It even learns to distinguish between healthy lines arising from element crossings or different element sides and potentially unhealthy cracks. If looking at the false positives and false negatives, one can see that the transition between the two classes is fluid and cannot be established beyond doubt even for a human inspector.

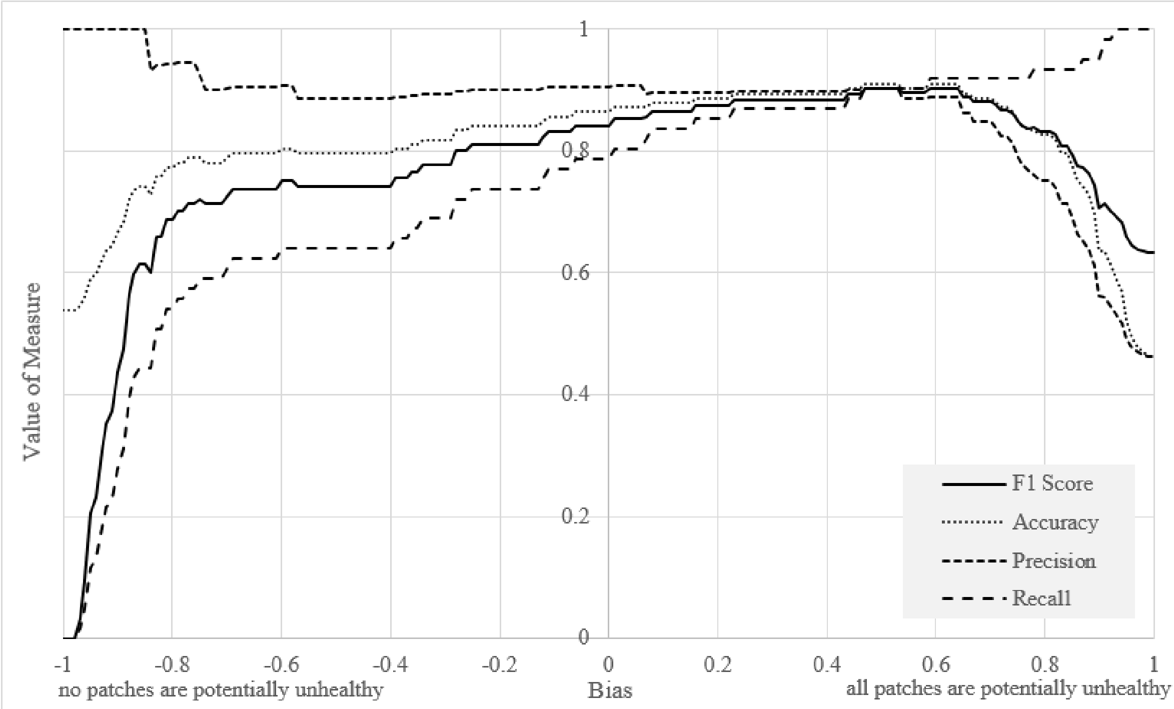

Figure 8: Analysis of the classification stability with respect to a bias towards one class.

A second experiment aimed at measuring the performance of the presented method versus the existing work in this field. McRobbie (2007) evaluated different metrics and metric combinations. Replicating the quadtree decomposition metric was not possible because it is not fully documented how the quadtree was built and which metric was included into the texture classification. The authors did not expect this to have a major impact on the classification results as it did not outperform the other metrics by far. Entropy was defined as in Formula 9 where p is the relative histogram counts of pixel values.

$$Entropy = -\sum_i(p_i \cdot log(p_i)) \qquad (9)$$

Figure 10 shows the results in comparison. The use of the same bias presented in the first experiment and different thresholds for the measures StdDev, Mean, Entropy, Metrics Combined, and Flip a Coin allows to contrast TNR and FNR as a continuous curve. Consequently, it can be decided which FNR or likelihood of missing a defect is allowed. This then gives the necessary bias or threshold and the corresponding TNR or reduction of search space. The horizontal axis represents the false negative rate

(FNR), which in this scope is the likelihood of missing a defect patch. The vertical axis represents the true negative rate (TNR) or the reduction of search space. This TNR is the decisive number to understand the potential of this method as it illustrates how much of an area can be skipped for a manual inspection. Out of the existing work, standard deviation and entropy worked best on the Cambridge Bridge Inspection Dataset. However, the newly presented method outperforms existing work considerably by achieving a reduction of search space of 90.1% at a risk of missing a defect patch of 8.2% at the maximal F_1 score determined in the first experiment. The actual risk of missing a defect is even lower if considering that a defect consists of multiple image patches.

Finally, a demonstration qualitatively examines the accuracy on an example surface texture where unseen defect samples were added using the image editing software Gimp. The surface texture is from a bridge column and has a surface size of 10.15 square meters. The image representing the surface texture has a resolution of 5,485 x 10,888 pixels and is manually enriched with spalling, efflorescence and a crack. Figure 11a shows the texture with defects and in red the outlined classification results. An overlap of 5/6 was used to achieve a high spatial resolution regarding the classification results and to be independent of the defect location within a single patch. The optimal degree of overlap was not further investigated because only a few reconstructed surface textures were available. The method detects all three example defects correctly. Only three areas are misclassified as false positive. Out of these two are the top and bottom part of the column, where stones, some asphalt parts of the street and grass is present in the image. The third part is the label that is present on this column to identify the bridge (such labels are not yet part of the training or evaluation dataset. They could be added to the dataset. The label is actually painted on the column and serves as bridge mark). The four other false positives are small and insignificant; three arise from a change of texture appearance due to previous maintenance work, one arise from texture reconstruction artefacts. It should be pointed out that all three defects are correctly detected and no defect has been missed. In addition, the size, shape and location of all three defects was detected correctly. Figure 11b shows the classification result in form of a normalized heat map. It visualizes the stability of the detection as all defect areas are clearly identifiable and differ considerably from the other texture. Figure 11c shows a close-up of the crack area, which is marked by the blue rectangle in Figure 11a. It is especially interesting to look at and compare the lines originated from concrete formwork and cracks. The presented method is able to distinguish between those two based on their unique appearance. Figure 11d illustrates that this does not depend on the absolute intensity change. It shows the cross-section intensity diagram for the formwork mark and the crack. The formwork mark has a greater footprint in the profile than the crack in both the absolute intensity value as well as in the spatial extent. Still, the trained model is able to distinguish and correctly classify both based on the different characteristics. Figure 11e shows the 3D view of the highlighted defects in our prototype implementation.

True Positives (Subset)

False Negatives

True Negatives (Subset)

False Positives

Figure 9: Example concrete patches, labels and classification results from Cambridge Bridge Dataset

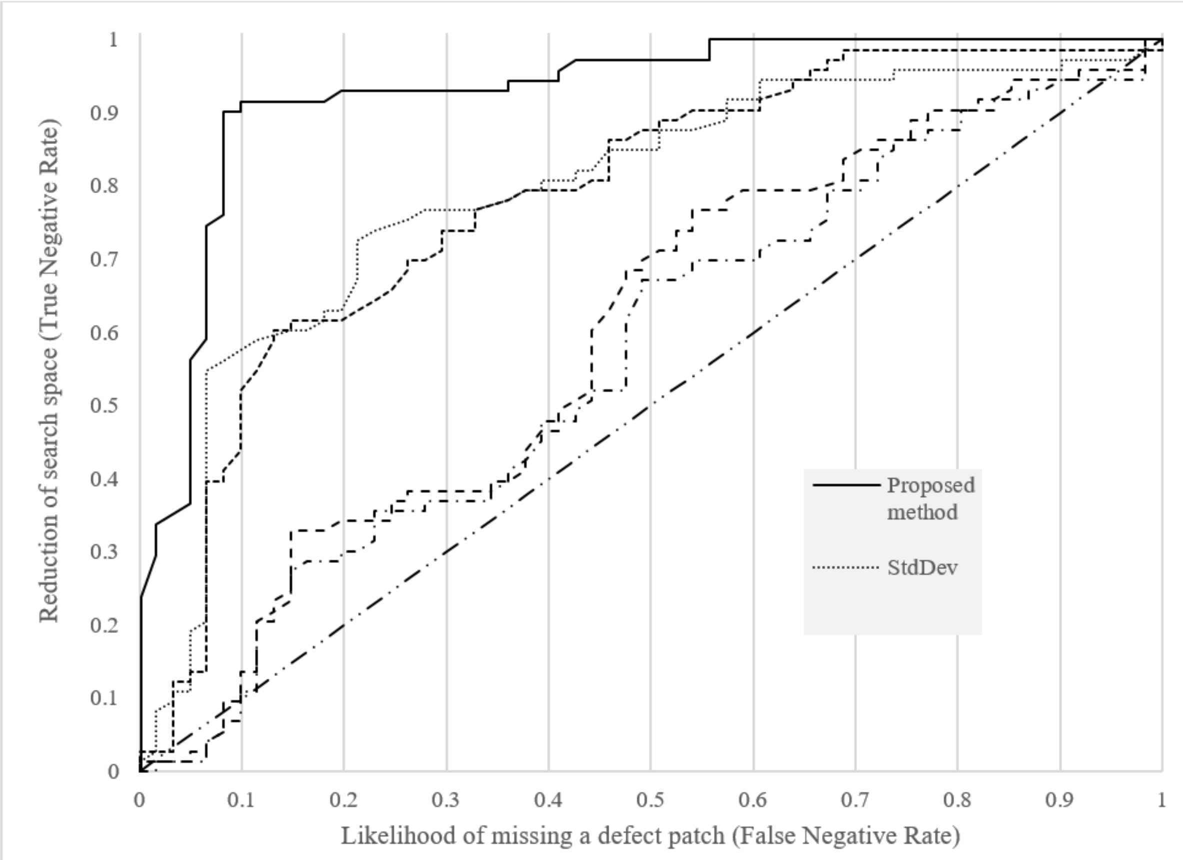

Figure 10: Comparison of presented method vs. existing methods regarding their reduction of search space and the likelihood of missing a defect.

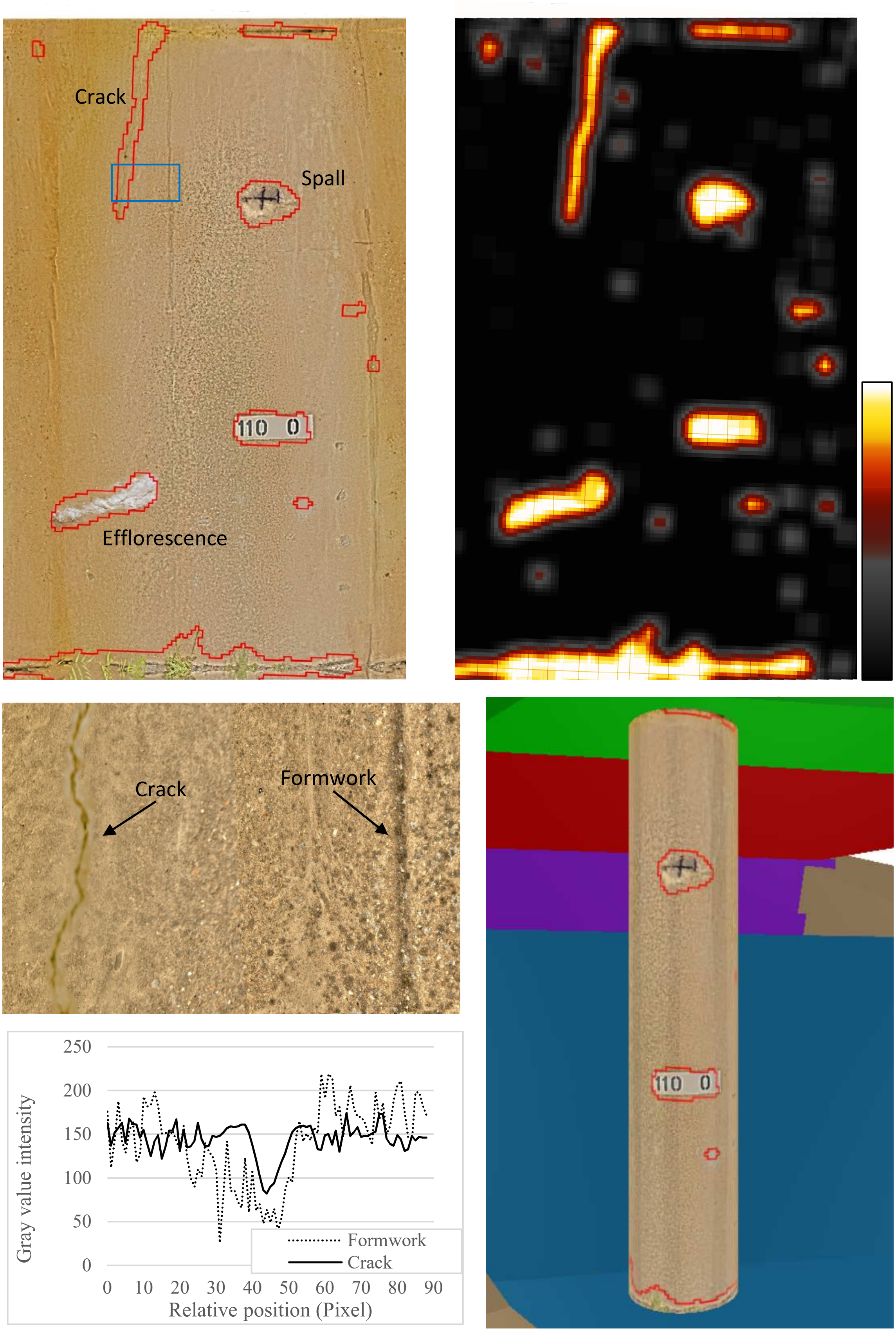

Figure 11: (a) Surface texture with manually added defect examples and overlaid classification results, (b) classification scores, (c) close-up comparison of a crack and formwork markings, (d) grey value intensity of a crack and formwork markings, (e) 3D view of as-is geometry with texture and defect highlight

4. Conclusion

The current practice of manual visual bridge inspection suffers from limitations such as inefficiency and subjectivity. Multiple efforts have been made to automate this task by automatically detecting specific defect types, mostly cracks. However, a variety of defects must be detected simultaneously, and detecting a subset of defect types does not solve the problem. More importantly, defects can appear in many different forms, colours, shades and textures. The existing methods do not generalize well with respect to varying defect classes and/or concrete appearance.

In this paper, the authors presented a method to automatically identify regions of interest in order to reduce the inspection space to areas which can then subsequently be inspected by a human engineer or by an automated defect classifier. This is done by inverting the problem from detecting potentially unhealthy concrete to detecting healthy concrete. A sliding window approach splits the surface texture in smaller chunks, such that it can be processed by a state-of-the-art classifier without losing small details such as cracks. A deep convolutional neural network is trained to detect healthy concrete. Classification results are merged to highlight potentially unhealthy areas directly on the element. This way, search space for inspectors is reduced to the areas which are not classified as healthy.

This approach can reduce the surface area that an inspector has to inspect by 90.1% with having a risk of missing a defect patch of 8.2%. It is assumed that a reduction of surface area has a proportional influence on the inspection duration. The per-defect failure rate is even lower based on the assumption that a defect is depicted in multiple patches. A bias towards the potentially unhealthy class enables to determine how much risk is acceptable to the cost of limiting the search space reduction. The authors have shown that the presented method is able to outperform existing methods for detecting potentially unhealthy areas for the scope of bridge inspection.

The contribution of this work is the process of slicing and merging high resolution bridge texture into patches, such that they can be processed with a state of the art image classifier. Our method has the benefit of not depending on multiple, hard to determine parameters and does not depend on handmade features. The method works end-to-end, taking the raw image data as input and directly outputting surface texture with potentially unhealthy areas highlighted. As with all machine learning approaches, the limitation of this method is that it is hard to understand how the classifier comes to a decision. The relevant feature vectors are trained automatically and are difficult to interpret as a human. Consequently, the classification reliability highly depends on the quality of the training data. This is particularly challenging for the scope of this work as there is no distinct definition of what is considered as potentially unhealthy and what is not. Different inspectors would label patches differently. This, however, is a reasonable weakness as controversial patches can be labelled as suspicious to be on the safe side. Increasing the size and diversity of the training dataset (variety of inspectors, agencies and countries) would result in a more representative dataset.

The presented method for automatically detecting potentially unhealthy areas can help to increase inspection efficiency by reducing the search space for a bridge inspector and guiding the inspector directly to the regions of interest. This way, the risk of missing a defect can be reduced. This will help to improve the overall quality of bridge condition information and hence will help to improve the cost-to-benefit ratio for transportation maintenance operations. It is important, however, to emphasize that this method does not solve the overall problem. It is just one component in the complex process of bridge inspection which needs to be reviewed and adjusted, continually and methodically. Only this way the overall inspection data quality, integrity and efficiency can be improved and technological advances in the field of civil engineering and computer science can be utilized. This leads to a fundamental change in the operation principles of the inspectors. More accurate and up-to-date condition information can help eliminate bridge maintenance backlogs while enabling municipalities to better ascertain road network service quality.

Multiple problems must be overcome in order to have a fully automated inspection solution. This is foremost the data collection and pre-processing task. An applicable technique to fully-automate or even semi-automate data collection of all relevant bridge element surfaces does not exist. This concerns and includes multiple disciplines, such as the sensor hardware (how is sufficient surface resolution achieved), sensor registration and actuation (using a drone, robot, handheld device) and legislation (major restrictions exist to fly drones close to bridges).

Data Access

The Cambridge Bridge Inspection Dataset supporting the findings of this study is available at University of Cambridge research repository with the identifier doi:10.17863/CAM.13813 (Huethwohl 2017)

Acknowledgements

We thank the representatives from Atkins UK, the U.S. Federal Highway Administration and Georgia DoT for supporting this research with parts of their bridge inspection image stock.

Funding

This work is partly funded by Trimble Inc. and by the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 31109806.0007. SeeBridge is co-funded by Funding Partners of the ERA-NET Plus Infravation and the European Commission. The Funding Partners of the Infravation 2014 Call are: Ministerie van Infrastructuur en Milieu, Rijkswaterstaat, Bundesministerium für Verkehr, Bau und Stadtentwicklung, Danish Road Directorate, Statens Vegvesen Vegdirektoratet, Trafikverket – Trv, Vegagerðin, Ministere de L'ecologie, du Developpement Durable et de L'energie, Centro para el Desarrollo Tecnologico Industrial, Anas S.P.A., Netivei Israel – National Transport Infrastructure Company Ltd. and Federal Highway Administration USDOT.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X., and Brain, G. (2016). “TensorFlow: A System for Large-Scale Machine Learning.” *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation*, Savannah, GA, USA, 265–283.

Alex Alemi. (2016). “Improving Inception and Image Classification in TensorFlow.” <https://research.googleblog.com/2016/08/improving-inception-and-image.html> (Oct. 19, 2017).

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2015). “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation.”

Bosc, M., Heitz, F., Armspach, J.-P., Namer, I., Gounot, D., and Rumbach, L. (2003). “Automatic change detection in multimodal serial MRI: application to multiple sclerosis lesion evolution.” *NeuroImage*, 20(2), 643–656.

Cireşan, D., Meier, U., and Schmidhuber, J. (2012). “Multi-column Deep Neural Networks for Image Classification.”

Collins, R. T., Lipton, A. J., and Kanade, T. (2000). “Introduction to the Special Section on Video Surveillance.” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22(8).

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). “Dermatologist-level classification of skin cancer with deep neural networks.” *Nature Publishing Group*, 542.

Golparvar-Fard, M., Balali, V., and de la Garza, J. M. (2015). “Segmentation and Recognition of Highway Assets Using Image-Based 3D Point Clouds and Semantic Texton Forests.” *Journal of Computing in Civil Engineering*, 29(1), 4014023.

Guo, W., Soibelman, L., and Garrett, J. H. (2009). “Automated defect detection for sewer pipeline inspection and condition assessment.” *Automation in Construction*, 18(5), 587–596.

Huethwohl, P. (2017). “Cambridge Bridge Inspection Dataset.” <https://doi.org/10.17863/CAM.13813>.

Huethwohl, P., Armeni, I., Fathi, H., and Brilakis, I. (2017). “Gygax Construction IT research platform for 2D & 3D.”

Hüthwohl, P., Brilakis, I., Bormann, A., and Sacks, R. (2017). “Integrating RC bridge damage data into BIM models.” *Structural Health Monitoring*.

Hüthwohl, P., Lu, R., and Brilakis, I. (2016). *Challenges of bridge maintenance inspection*. *16th International Conference on Computing in Civil and Building Engineering*.

Icaza, M. de, Dorokhov, Souza, C., Syme, D., Pantyukhin, A., Leibowitz, M., and Kernahan, A. (2017). “TensorFlowSharp: TensorFlow API for .NET languages.” <https://github.com/migueldeicaza/TensorFlowSharp> (Apr. 5, 2018).

Jahanshahi, M. R., Asce, A. M., Jazizadeh, F., Asce, S. M., Masri, S. F., Asce, M., and Becerik-Gerber, B. (2013). “Unsupervised Approach for Autonomous Pavement-Defect Detection and Quantification Using an Inexpensive Depth Sensor.” *Journal of Computing in Civil Engineering*, 27(6), 743–754.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). “Caffe: Convolutional Architecture for Fast Feature Embedding.” *Proceedings of the ACM International Conference on Multimedia - MM ’14*, ACM Press, New York, New York, USA, 675–678.

Koch, C., Georgieva, K., Kasireddy, V., Akinci, B., and Fieguth, P. (2015). “A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure.” *Advanced Engineering Informatics*, 29(2), 196–210.

Kong, J. S., Asce, M., Frangopol, D. M., and Asce, F. (2003). “Life-Cycle Reliability-Based Maintenance Cost Optimization of Deteriorating Structures with Emphasis on Bridges.”

Lerma, J. L., Cabrelles, M., and Portalés, C. (2011). “Multitemporal thermal analysis to detect moisture on a building façade.” *Construction and Building Materials*, Elsevier, 25(5), 2190–2197.

Long, J., Shelhamer, E., and Darrell, T. (2014). “Fully Convolutional Networks for Semantic Segmentation.” Computer Vision and Pattern Recognition.

Lu, D., Mausel, P., Brondízio, E., and Moran, E. (2004). “Change detection techniques.” *International Journal of Remote Sensing*, Taylor & Francis Group , 25(12), 2365–2401.

Lu, R., and Brilakis, I. (2017). “Recursive Segmentation for As-Is Bridge Information Modelling.” *Lean and Computing in Construction Congress - Volume 1: Proceedings of the Joint Conference on Computing in Construction*, Heriot-Watt University, Edinburgh, 209–217.

Marks, R. J. (1991). *Introduction to Shannon Sampling and Interpolation Theory*. Springer Texts in Electrical Engineering, Springer New York, New York, NY.

Matsumoto, M., Mitani, K., and Catbas, F. N. (2012). “Bridge Assessment Methods using Image Processing and Infrared Thermography Technology.”

McRobbie, S., Lodge, R., and Wright, A. (2007). *Automated Inspection of Highway Structures Stage 2 - PPR 255*.

McRobbie, S., Woodward, R., and Wright, A. (2011). “Visualisation and display of automated bridge inspection results - PPR530.” *Visualisation and display of automated bridge inspection results*, 1(1), 1–28.

Murdock, K. (2008). *3ds Max 2009 bible*. Wiley.

Phares, B. M., Washer, G. A., Rolander, D. D., Graybeal, B. A., and Moore, M. (2004). “Routine Highway Bridge Inspection Condition Documentation Accuracy and Reliability.” *Journal of Bridge Engineering*, ASCE, 9(4), 403–413.

Radopoulou, S. C., and Brilakis, I. (2016). “Improving Road Asset Condition Monitoring.” *Transportation Research Procedia*, Elsevier, 14, 3004–3012.

Spencer, F. W. (1996). *Visual Inspection Research Project Report on Benchmark Inspections*.

Stent, S., Gherardi, R., Stenger, B., and Cipolla, R. (2015). “Detecting Change for Multi-View, Long-Term Surface Inspection.” *British Machine Vision Conference*.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. (2016). “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning.”

The Highways Agency. (2007). *Inspection manual for highway structures.* TSO.

Valença, J., Puente, I., Júlio, E., González-Jorge, H., and Arias-Sánchez, P. (2017). “Assessment of cracks on concrete bridges using image processing supported by laser scanning survey.” *Construction and Building Materials*, 146, 668–678.

Xie, F., and Levinson, D. (2011). “Evaluating the effects of the I-35W bridge collapse on road-users in the twin cities metropolitan region.” *Transportation Planning and Technology*, 34(7), 691–703.