Defect detection by image processing

Technical Field

[0001] This disclosure generally relates to methods and systems for detecting defects in components based on image processing operations.

Background

[0002] This background description is provided for the purpose of generally presenting the context of the disclosure. Contents of this background section are neither expressly nor impliedly admitted as prior art against the present disclosure.

[0003] Defects such as cracks or scratches in products or infrastructure artefacts are a significant problem and a cause of concern in many manufacturing domains. Early detection of such defects is important for safe and efficient operations while maintaining the quality standards of the produced goods. Accordingly, defect detection is an important problem relevant for various applications such as infrastructure surface inspection, product quality monitoring and airplane/ship surface inspection. Many computer vision-based defect detection methods including deep learning based methods been proposed to reduce labor cost and improve efficiency. Deep learning based methods of defect detection rely on existing datasets of example defects to train neural networks to detect defects. However, unlike tasks such as conventional object detection, the patters in which defects can present have a large degree or variability. For example, the differences in features between a cat and a dog are far greater than the differences in two defect related image artefacts. While conventional deep learning techniques may provide good performance and accuracy with problems such as distinguishing between cats and dogs, or even species of cats and dogs, the conventional deep learning techniques may not be sufficient to distinguish between two defect related image artefacts that present a much greater degree of variability.

[0004] Existing datasets for training neural networks for defect detection usually comprise low resolution images exemplified in images 100 (with defect 102), 120 (with defect 132) of Figure 1. Therefore, conventional deep learning based defect detectors are trained to handle such low resolution images and defects that may be discernible in such images. However, in industrial applications, such as defects detection on raw images

from surveillance cameras or camera integrated in a manufacturing line, resolution of images to consider may be much higher. The higher resolution images may comprise 2,560 x 1,920 pixels, 6,000 x 4,000 pixels or more. The higher resolution images comprise irregular defects and have low Signal-to-Noise-Ratio as illustrated in images 110 (defect 112) and 130 (defect 142) of Figure 1. Further, images of product surfaces may have odd aspect ratio or unconventional aspect ratios. Applying a detector such as a deep learning based detector pre-trained on low resolution images and roughly square images to high-resolution or odd aspect ratio images provides undesirable and inaccurate outcomes. It is desirable to provide more efficient, versatile and accurate systems and methods for defect detection or at least provide an alternative to convention defect detection systems and methods.

Summary

[0005] In one embodiment, the present disclosure provides a system for defect detection in images, the system comprising:

one or more processor(s);

a memory comprising instructions executable by the processor(s) to:

receive a target image for defect detection;

process the target image using a reinforcement learning agent (RL agent) to identify a plurality of coarse defect regions in the target image;

process each coarse defect region using an object detection model (OD model) to identify a sub-region of the coarse defect region, the sub-region corresponding to a defect;

wherein the OD model is trained using a low-resolution image dataset; and

the RL agent is trained using a high-resolution image dataset.

[0006] In some embodiments, processing the target image using the RL agent to identify the coarse defect region comprises: region selection operation; and region refinement operation.

[0007] In some embodiments, the RL agent is trained to optimize losses between the region selection operation and the region refinement operation.

[0008] In some embodiments, the RL agent is trained to perform coarse defect region selection operation based on feedback from the OD model.

[0009] In some embodiments, the RL agent is trained in a first phase to minimize losses for the defect region selection operation.

[0010] In some embodiments, the RL agent is trained in a second phase to minimize losses for both defect region selection operation and region refinement operations.

[0011] In some embodiments, the RL agent is trained using a dynamic threshold setting mechanism for the RL agent's reward function.

[0012] In some embodiments, the RL agent is trained using the reward function $R(s,a) = \begin{cases} 1 + IOU_{diff} & if\ IOU_{det} \geq IOU_{th} \\ 0 & else \end{cases}$

wherein IOUdet is the Intersection Over Union(IOU) between ground truth defect and detected defect, IOUdiff is the difference of IOUdet before and after region refinement, and IOUth is the threshold.

[0013] In some embodiments, the defect comprises a crack or a scratch on a surface.

[0014] In some embodiments, the high-resolution image dataset is a dataset of high-resolution dataset of unannotated images.

[0015] Some embodiments relate to a method for defect detection in images, the method comprising:

receiving, at a processor, a target image for defect detection;

processing the target image using a reinforcement learning agent (RL agent) provided in a memory accessible to the processor to identify a coarse defect region in the target image;

processing each coarse defect region using an object detection model (OD model) provided in the memory to identify a sub-region of the coarse defect region corresponding to a defect;

wherein the OD model is trained using a low-resolution image dataset; and

the RL agent is trained using a high-resolution image dataset.

Brief Description of the Drawings

[0016] Some embodiments of systems and methods for defect detection, in accordance with present disclosure are described with reference to non-limiting examples illustrated in accompanying drawings in which:

[0017] Figure 1 illustrates images of defects in various resolutions;

[0018] Figure 2(a) illustrates a block diagram of an architecture for training a reinforcement learning agent;

[0019] Figure 2(b) illustrates a block diagram of another architecture for training a reinforcement learning agent;

[0020] Figure 2(c) illustrates a block diagram of another architecture for training a reinforcement learning agent;

[0021] Figure 3 illustrates a series of images illustrating various stages of defect detection;

[0022] Figure 4 illustrates a high-resolution image with bounding boxes around defect features;

[0023] Figure 5 illustrates a block diagram of an architecture of a policy network for defect detection;

[0024] Figures 6 and 7 illustrates images with defects and results of defect detection using the disclosed and comparative methodologies;

[0025] Figure 8 illustrates a block diagram of a system for defect detection; and

[0026] Figure 9 illustrates a flow chart of a method for defect detection.

Detailed Description

[0027] Embodiments relate to systems and method for defect detection by image processing operations. The defects may relate to defects in products such as electronic products, mass produced goods, furniture etc. The defects may alternatively relate to defects in infrastructure such as buildings/parts of buildings, bridges/parts of bridges, oil rigs, vehicles, airplanes etc. More generally the defects may also relate to defects in

surfaces such as defects in the form of cracks, scratches, unexpected irregularities etc. The embodiments may be incorporated in an industrial production line or may be a part of an inspection system for defection of defects. The embodiments may be used to perform defect detection in the field (i.e. in close vicinity to the object being inspected). Alternatively, the embodiments may receive images from remote imaging systems and perform defect detection away from the field based on received images.

[0028] Figure 8 illustrates a block diagram of an exemplary system 800 for defect detection. The system 800 comprises one or more processor 810, a memory 820 that store program code that is executable to perform defect detection. The system 800 may receive images captured by a camera 840 for detection of defects in the received images. The memory 820 comprises program code to implement a reinforcement learning agent/model 822 and an object detection (OD) model 824. The embodiments perform efficient defect detection in images using a combination of specifically RL agent and the OD model. The flowchart of Figure 9 illustrates a method of defect detection executable by the system 800.

[0029] The embodiments perform a hierarchical/cascaded analysis of images for defects using the RL agent to identify at least one coarse defect region in an image. Some embodiments comprise a processing pipeline for applications that identify defects in high-resolution (HR) images using a pre-trained deep learning (DL) object detection model without need for extra image annotation work and re-training of DL model on HR images. HR images may comprise images of a resolution of 1920 x 1080 pixels or greater.

[0030] The DL model of some embodiments may be pre-trained on low resolution (LR) images. Given the accuracy of conventional DL models relies on the availability of a large volume of training data, most conventional DL models are adapted to process low resolution images with a relative high defect background ratio (DBR). The DBR is the ratio of the region of the image featuring a defect and the remaining region of the image featuring the background. With relatively higher DBR values in the images in the training set, the conventional DL models may produce acceptable defect detection results.

[0031] The RL agent of some embodiments is trained using HR images with lower DBR to predict defect comprising region in the images. At step 910 of Figure 9, the system 800 receives a target image for defect detection. Step 920 comprises processing of the target image by the RL agent to identify one or more coarse defect regions. The RL agent performs two steps as part of step 920. The first step defective region selection

(step 922 of Figure 9) comprises selection of one or more candidate regions within the target image. Each candidate image provides a potential starting point the subsequent step of region refinement (step 924). As part of step 922, multiple regions of the HR image are sampled on the target HR image based on its dimension. This may comprise defining a series of grids on the image to segment the image into multiple regions. In some embodiments, the grids may be uniform and each sampled region may be identical in size. In other embodiments, the grids may be non-uniform and may be optimized depending on the position or field of view capturing the images.

[0032] The action space for defective region selection task of the RL agent is determined by reference to the series of grids. The RL agent is trained to predict defective region index (identifier of one of the series of grids). The RL agent does so based on detection results (feedback) from DL object detector which forms part of the environment of the RL agent. For example, the DL object detector may process each region of the target image and generate a score indicative of presence of a defect. The scores serve as feedback for the RL agent to optimize the selection of one or more regions (candidate defect regions) most likely comprising a defect.

[0033] To improve the DBR in the selected candidate defective regions, the RL agent is further trained to perform region refinement. Region refinement comprises optimization of the location of the candidate defect regions to cover or encompass more accurately the defects present in the target image. The refinement operations may comprise movement of the selected candidate region up, down, left or right within the target image. The refinement operation may also comprise scaling operations including expansion or contraction of the candidate defect regions. The refinement operation may also comprise a combination of two or more selected candidate defect regions into a combined candidate defect region.

[0034] The selected and refined regions may be referred to as coarse defect regions. A coarse defect region is a region of an image that includes at least a part of a defect or expected defect in an image. The coarse defect region may contain parts of the image that surround the defect/expected defect. The defect/expected defect may not be centered in the coarse defect region. Figure 2(a) illustrates an example of a selected defect region 212 that is subsequently refined to the region 214 to provide a better coverage of the potential defect and improve the SNR of the image segment that may be treated as the coarse defect region.

[0035] The coarse defect regions are evaluated using the OD model 824 at step 930 of the flowchart of Figure 9 to generate final defect regions in the target image. In target images without any defects, no final defect regions are identified. The final defect regions may be mapped back to the HR images. Some embodiments advantageously reduce the need for annotation of HR images for specifically training an OD model for detecting defects in HR images. Annotation of images is a time and labor intensive process that is prone to errors. In experiments on HR datasets of cracks and scratches, the embodiments demonstrated state-of-the-art performance with 0.976 and 0.965 F_1 score for crack and scratch, respectively.

[0036] Some embodiments incorporate a dynamic threshold setting mechanism to improve convergence of RL agent training, while improving detection performance. Some embodiments also incorporate progressive optimization between defective region selection loss and regions refinement choice loss to reduce their influence on each other. As the RL agents performs both region selection and region refinement, it is beneficial to strike a balance between the region selection and region refinement operations during the training of the RL agent to obtain improved defect detection results. Some embodiments do so by progressively optimizing the two different losses during training as described in the subsequent sections.

The Configuration of Hierarchical Pipeline

[0037] An architecture of the hierarchical defect detection is illustrated in Figure 2(a). The embodiments contain two components: OD model 824 and RL agent 822. The OD model may be pre-trained to detect defects on the LR images. This is further illustrated in Figure 2(b) and 2(c). The OD model serves two purposes: (1) in RL agent training stage, it acts as an environment to provide feedback based on action execution of the RL agent on each state; and (2) in inference stage, it provides final defect detection results by processing the coarse defect regions selected and refined by the RL agent.

[0038] The RL agent is trained by optimizing the parameters of a policy network of the RL agent to maximize accumulated actions reward from its simulated environment. In the design of the RL agent's conception, the following different stages were considered: design of the RL policy network, state representations of each action of the RL agent, design of a reward function and loss optimization between region selection and region refinement.

OD Model Training

[0039] The DL object detector may be pre-trained on a dataset of lower resolution images. In training of DL model of some embodiments two low resolution (LR) datasets are employed: a data set of 256x256 pixel images and pavement cracks dataset with 227x227 pixel images. 5,000 images from former and 3408 from later dataset were randomly selected to obtain a total of 8,408 training images.

[0040] The DL model of some embodiments was specifically trained to detect scratches. In some embodiments, the DL model was trained to detect both scratches and cracks. To train the DL model to detect scratches, 300 scratch images from the NEU surface scratch database (200x200 pixel images) were used. As part of the preparation of the training dataset, data augmentation was performed to enrich the data. The data augmentation comprised rotation of images, incorporation of Gaussian blur etc. The augmentation provided a training set of 3600 images.

[0041] The OD model may be implemented using Faster R-CNN, YOLO or SSD or other conventional frameworks for object detection. The OD model of some embodiments incorporated Resnet-101 are a backbone. In some embodiments, a training dataset from training the OD model is prepared by a semi-automatic method applied to lower resolution images. In the training images, a curve may be overlaid on the crack or scratch, and a random subset (for example 10%) of the pixels of the curve may be selected as seed points. Ground truth patches may be generated by reference to the subset of pixels. Each patch may have a predefined dimension, for example 32x32 pixel patches. Least square fitting may be applied to the ground truth patches to annotate the training images.

[0042] The OD model is trained using the prepared training dataset to detect scratches and cracks in the lower resolution image dataset. The detection results from the trained OD model are series of bounding boxes of potentially defective patches as illustrated in Figure 3(b) which were obtained by processing the image of Figure 3(a). After the generation of the series of bounding boxes around potential defects, contour estimation is performed as illustrated in Figure 3(c). The estimated contour extends along a length of the extended defect spread across multiple bounding boxes. Based on the estimated contour, a mask is generated as illustrated in Figure 3(d) along the extended defect outline. In some embodiments, the mask region of extracted boundaries from detected bounding boxes is presented as an output of the OD model during both RL agent training and defect inference stages.

RL Agent Optimization

[0043] The defective region selection and refinement task is characterized as a control problem in a Markov Decision Process (MDP). This process is represented in a tuple (S, A, P, R, γ). Given continuous state-action space, at time step *t*, the state is $s_t \in S$ and the action is $a_t \in A$, the transition function from s_t to s_{t+1} with action a_t is represented as $P(s_{t+1}|s_t, a_t)$, the immediate reward after the transition is denoted as $r_t=R(s_t, a_t)$. $\gamma \in [0,1]$ is the discount factor used to balance current and future reward. Given state s_t, action a_t selection policy is $\pi(a_t|s_t)$. The next state s_{t+1} can be computed according to current state s_t and action section policy π of s_t in Eq. (1).

$$s_{t+1} = P(s_t, \pi(s_t)) \quad (1)$$

[0044] Given a state, the target of defective region selection is to find an optimal action selection policy π, which results in the selected region comprising the maximum possible part of a potential defect if not the entire defect. In essence, the optimal actions are intended to increase the signal to noise ratio of the selected region. To do so, the RL agent is tasked to achieve maximum accumulated reward. In some embodiments, a model free Q-learning RL framework is selected as RL algorithm. Alternative RL frameworks may be selected to achieve the same goals. Q-learning learns the action-value function Q(s, a) and relies on deep Q-learning network (DQN) to select an action at a given state. By estimating action-value (Q-value), the optimal policy can be determined by Eq. (2)

$$Q^{\pi}(s_t, a_t) = E[R_t|s_t, a_t] \quad (2)$$

[0045] And the output Q-value of DQN on action *a* is computed by combining state value and action type advantage over average value, which is shown in Eq. 3

$$Q(s, a;\ \theta) = V^s(s; \theta^f, \theta^s) + (A(s, a; \theta^f, \theta^a) - \frac{1}{N}\sum_{a' \epsilon A} A(s, a'; \theta^f, \theta^a)) \quad (3)$$

where V^s is the state value, θ^f is the parameters of common part of feature extractor (policy network), θ^s and θ^a represent parameters of state value and action, respectively. *N* is number of regions to be selected.

[0046] The task of RL agent optimization comprises an action-value (Q-value) function optimization: $Q(\theta) \rightarrow Q(\theta^*)$, which is the target of RL agent training. The parameters θ of DQN are optimized by minimizing loss which is mean square error between target Q-value (Q_t) and current Q-value (Q_e), the Q_t and loss are defined in Eq. (4)

$$\begin{cases} Q_e = Q(S_{t-1}, a_t; \theta) \\ Q_t = r_t + \gamma \max_{a_{t+1}} Q(S_t, a_{t+1}; \theta) \\ L(\theta) = [Q_t - Q_e]^2 \end{cases} \quad (4)$$

[0047] After training, the optimal parameters θ* are obtained for action selection policy. An action selection policy can be expressed as in Eq. (5)

$$a_t = arg \max_{a \epsilon A_t} Q(S_{t-1}, a; \theta^*) \quad (5)$$

To train the RL agent, five key parts of RL are defined:

- Environment: the physical world in which the agent interacts and obtains feedback
- State representation: input of the policy network for action decision making
- Action space: the number and types of actions allowed for the RL agent
- Reward function: the way to compute feedback from environment based on selected action given current state
- Policy network: a neural network to extract features from states and output agent action selection

State Representation and Action Space

[0048] The RL agent is trained to perform region selection and region refinement in a cascaded manner to obtain coarse defect regions. The RL agent accordingly has two state representations, one for each stage as illustrated in Figure 2(b). The state representation for the region selection receives an input image (target image). A region selection decision is made after feeding the target image to an OD model to select regions potentially comprising defects. The selected regions form part of the state that serve as an input for the region refinement operation.

[0049] The action space for defective region selection depends on the dimension the target image. Since the target image may comprise high resolution images, the high resolution images may be fragmented in a grid like pattern, wherein each segment of the grid serves as part of the action space for region selection. Given the OD model is trained to perform object detection using a training dataset of relatively fixed dimensions, the action space for region selection may be defined with reference to the dimensions of the images used for training the OD model. In some embodiments, the layout of the region on target high resolution images for region selection can be determined by Eq. (6).

$$\begin{cases} a * M - b * (M - 1) = W \\ a * N - b * (N - 1) = H \\ b = \frac{a}{8} \end{cases} \quad (6)$$

where *a* is the width of LR images used for training the OD model and *b* is the overlap between two adjacent regions; *M* and *N* are number of regions in horizontal and vertical directions of HR images; *W* and *H* are width and height of HR images. The action space is can be determined as *M*N*. Figure 2(c) illustrates an example target image 240 that has been fragmented into 12 segments, each segment serving as part of an action space for selection by RL agent. In Figure 2(c), the RL agent select region index 7 as the potentially defect comprising region.

[0050] The action space for region refinement comprises five actions: *up*, *down*, *left*, *right* and *no_movement*. Once a refinement action on a region is selected, the bounding boxes of the region may be shifted toward a direction considered as a most optimum direction of movement by the RL agent. In Figure 2(c) the RL agent selected a down movement to refine to region 242 as one that provides a better coverage of the potential defect and a higher SNR.

Progressive Loss Minimization

[0051] The policy network of the RL agent needs to be optimized on two actions: defective regions selection and regions refinement. Multiple actions policy optimization in RL can be formulated as multi-loss minimization where all actions are equal and independent. However, in the disclosed systems, the region selection happens before region refinement, as illustrated in Figure 2(b). Moreover, the region refinement is only meaningful when the selected region is defective. The RL agent of some embodiments in trained to progressively minimize losses on the two actions in the training process. In the first *W* epochs (phase I), the embodiments optimize action policy on defective region selection by only minimizing loss on region selection. The region refinement prediction is set as *no_movement*. By doing so, after phase I training, the RL agent is able to make effective decision on defective region selection. In phase II of the training process both action policies are optimized by jointly minimizing losses on region selection and refinement. The loss function in phase I and phase II are defined in Eq. (7) and (8).

$$L^I(\theta) = [Q_t^{rs} - Q_e^{rs}]^2 \quad (7)$$

$$L^{II}(\theta) = [Q_t^{rs} - Q_e^{rs}]^2 + [Q_t^{rf} - Q_e^{rf}]^2 \quad (8)$$

$Q_t^{rs} - Q_e^{rs}$ refers to the loss associated with region selection. $Q_t^{rf} - Q_e^{rf}$ refers to the loss associated with region refinement. After the region selection and region refinement, a coarse defect region is identified by the RL agent (assuming a defect is present in the input target image).

Reward Function

[0052] Reward function R(s, a) is designed to encourage effective actions to improve the accuracy of defect detection. In some embodiments, the reward function is defined as:

$$R(s,a) = \begin{cases} 1 + IOU_{diff} & if\ IOU_{det} \geq IOU_{th} \\ 0 & else \end{cases} \quad (9)$$

where IOU_{det} is the Intersection Over Union(IOU) between ground truth defect and detected defect. IOU_{diff} is the difference of IOU_{det} before and after region refinement. The reward based on IOU_{diff} is defined for encouraging effective refinement movement. Considering progressive losses minimization, in the phase I, IOU_{diff} is 0 since there is no refinement movement in this phase. IOU_{th} is the threshold. Normally, it is set as a fixed value. Once IOU_{det} is larger than the value, a positive reward is returned, which indicates that the RL agent training on current state is finished successfully. Otherwise, no reward is allocated to the action.

[0053] Setting the threshold for reward return as fixed value may result in two undesirable consequences: (1) if the threshold is set at a relatively low value, the RL agent may finish training on an image too early and miss the most defective region; (2) if the threshold is fixed as a high value, there may be a chance that none of the predefined region (containing enough defects) meets requirement. Thus, the RL agent training on an image thus will be divergent. Taking image in Figure 4 as an example, if the threshold is set as <0.5, region A (bounding box 410) is accepted as a positive region, and the RL agent training will be terminated on the image once region A is selected. However, region A is not necessarily the optimal solution as region B (bounding box 420) provides a better coverage of defects. If the threshold is fixed as a high value, e.g., 0.9, no region is qualified to be a terminal region. Hence, the RL agent may miss defective region selection on the image. To address these undesirable outcomes, embodiments incorporate a dynamic threshold setting mechanism in the reward function. The dynamic threshold setting mechanism enables the trained RL agent to optimize the selection of coarse defect

regions in a target image such that a maximum possible portion of relevant defects are captured in the coarse defect regions after region selection and refinement. Algorithm 1 below illustrates an example of a dynamic threshold setting mechanism incorporated by some embodiments. In Algorithm 1, r stands for reward. The r value being equal to 1 indicates that all potential defects have been detected in an epoch iteration (image processing iteration). In the event a defect has been detected in an image iteration the dynamic threshold remains at T_{ini}. Otherwise, the $T_{dynamic}$ is progressively decremented over the epoch iterations. As the $T_{dynamic}$ is decremented over the epochs this improves the likelihood of finding all defects in an image. As illustrated in line 14, if r is not equal to 1 (i.e. all potential defects have not been detected) the $T_{dynamic}$ may be decremented as a function of the fraction of the image iteration vs the total number of training samples (i.e. e/K). As training progress through iterations, the $T_{dynamic}$ value is reduced to more optimally suit the dataset and improve the defect detection outcomes. In an alternative embodiment, the $T_{dynamic}$ value may be initiated with a smaller value that is incremented as training occurs.

Algorithm 1: Dynamic Threshold Reward Setting

1. Initial threshold for reward: T_{ini}
2. Total training epochs: E_{total}
3. Total training samples: S_{total}
4. Action space on an action: n;
5. Step size for threshold adjustment: ϱ
6. Frequency to check the threshold: K← S_{total} *n
7. Reward value RL agent obtained on all training samples in epoch k:{R_k}
8. **for** e←1 to E_{total} **do**
9. **if** e%K==0 **then**
10. **for** r in { R_k } **do**
11. **if** r==1 **then**
12. $T_{dynamic}$ ←T_{ini}
13. **else**
14. $T_{dynamic}$ ←T_{ini}-e/K*ϱ
15. **end**
16. **end**
17. **else**
18. **continue**
19. **end**

RL Agent Training

[0054] The RL agent is trained by optimizing the policy network in DQN framework. In embodiments where alternatives to the DQN framework are incorporated, the described training techniques may be adapted to suit the relevant framework to obtain similar outcomes. An architecture of the policy network is exemplified in Figure 5. The network may comprise 5 convolutional layers in which each has a filter with a kernel size of 5x5, stride of 1, padding of 2 and RELU as the activate function. Of the first three convolutional layers, each is followed by a max-pooling layer for feature maps down sampling. The output of the convolutional layers is 64x8*8 feature maps, which will be processed by a flatten and FC layer and converted to a 1D vector of 1024 dimension. Then, the three output layers are converted to three vectors, namely, region selection advantage, region refinement advantage and state value. At last, the state value will be combined with region selection and region refinement advantages respectively to form the region selection Q-value (output by neural network portion RS 510 in Figure 5) and the region refinement Q-value (output by neural network portion RF 520 in Figure 5).

[0055] In the training process of some embodiments prioritized experience replay techniques are employed to sample experience which is a tuple containing the current state, action, the next state and reward with high priority from a replay buffer. The sampling process happens after every action selection, and selected experiences are stored in a mini-batch to optimize the parameters of the DQN model. The incorporation of prioritized experience replay enables the RL agent to benefit from past optimization actions and reuse them as necessary. Some embodiments incorporate the Adam optimizer with a learning rate of 0.001, the mini-batch size is 64, a capacity of replay memory of 10,000; a discount factor for future reward of 0.99; a training epoch is 1000. These optimization parameters may be varied depending of the relevant dataset and application domain to obtain the most accurate defect detection results. The target Q value (Q_t) and evaluate Q value (Q_e) are randomly initialized. The ϵ-greedy policy, is adopted to allow the RL agent to exploit most of the time with a small chance of exploring. After every 40 episodes, the Q_t will be updated by Q_e. A detailed example of a DQN model training is shown in Algorithm 2 below.

Algorithm 2: DQN model training

Initialize prioritized replay memory ***D***
Initialize Q_e and Q_t with random parameters θ_e and θ_t
for e from 1 to *E* **do**
  Random select start state s
  Set done ← False
  while *not done* **do**
    if *Random* (0,1) <ϵ **Then**
      Select random action on a_t^{rs}
      If *e>w* **then**
        Select random action on a_t^{rf}
    else
      Predict a_t^{rs} using Eq. (5)
      If *e>w* **then**
        Predict a_t^{rf} using Eq. (5)
    Compute r_t using Eq. (9)
    If *success or failure* **then**
      done ←True
      s_t←NULL
    else
      Update s_t using Eq. (1)
    Store experience (s_t, a_t^{rs}, a_t^{rf}, r_t, s_{t+1}) in ***D***
    Sample mini-batch in ***D***
    for each experience in mini-batch **do**
      Calculate y^{rs} and y^{rf}
      y^{rs}=Q_t^{rs} in Eq. (4)
      y^{rf}=Q_t^{rf} in Eq. (4)
      if e < w **then**
        Update θ_t by gradient descent to minimize loss on a_t^{rs} in Eq. (7)
      else
        Update θ_t by gradient descent to minimize losses on a_t^{rs} and a_t^{rf} in Eq. (8)
  if e % 40 ==0 **then**
    $\theta_t \leftarrow \theta_e$
Return $\theta^* = \theta_e$

Evaluation

[0056] The embodiments were evaluated on two HR crack and scratch datasets. The crack dataset contains 1240 images with a resolution of 1920*2560, and the scratch dataset contains 1624 images with resolution of 800*4000. For each dataset, half of the images were used for RL agent training, and the other half were reserved for testing. If a defective region was selected and at least half of defect is detected by DL detector, the region is treated as a True Positive (TP). Otherwise, it is a False Negative (FN). If a region that does not contain defect but selected as defective region, it is called a False Positive (FP). Otherwise, it is a True Negative (TN). To make a comprehensive estimation

of the embodiments, the metrics Precision(P), Recall(R) and F1 score were calculated as:

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}, \quad F1 = \frac{2PR}{P + R}.$$

[0057] The embodiments were compared against three conventional methods, namely direct image scaling of HR images to LR images before defect detection (Method A), sliding window based analysis (Method B) and retraining of an OD model on HR images (Method C). The quantitative benchmarking results on crack and scratch datasets are tabulated in Tables 1 and 2. Based on the results the following observations follow:

(I) Method A may provide acceptable performance with F1 of over 0.8. The inferior consequence brought by directly scaling HR images to LR images in testing stage is that the visual features in the HR images after scaling are deformed, which leads to missed defects or false detection of defects not present.

(II) Method B achieves the highest Recall and lowest Precision due to it densely scan overlapped regions leading to many false alarms as well as longer evaluation time.

(III) Method C provides the lowest Recall on both datasets, especially on scratch dataset, the detector almost missed all defects. There are two possible reasons. (1) The backbone CNNs (i.e. VGG, ResNet) for the detector training framework is usually quite deep. For very small objects (i.e. crack in HR images), after going through the image processing pipeline (e.g., resizing the training images to a fixed dimension (i.e. 224 *224), convolution operations through many layers), these small objects are beyond the processing scope of the backbone CNNs. Therefore, they may be abandoned in the detector training process. (2) The dimensions of annotated bounding boxes of defects in OAR images (i.e. scratch images) become very odd after resizing the OAR images. This results in the dimensions of resized bounding boxes of defects to be out of scope of all normal anchor or grid size of detector training framework. Alternatively, customized anchor size can be investigated for particular objects, whereas this needs extra investigation work.

(IV) Methods according to the embodiments provided the highest Precision and F1 score.

[0058] Some visualized comparison results are shown in Figure 6 and 7. Detected defects are indicated in bounding boxes for the various methods. Defect detection by the disclosed embodiments identify a coarse defect region (for example regions 610, 620, 630, 640 and 650). Within the coarse defect regions, the cracks or scratches are identified

more precisely by the OD model. The more precise identification is illustrated through a plurality of bounding boxes that when considered together identify a defect extending within a coarse defect region.

[0059] From the visualized results in Figure 6 and 7, it is observable that Method A, defects are either missed or detected imprecisely. The ratio of the real defect to the detected regions are very low for Method A. This is because when the detector of Method A was trained on the LR dataset, the anchor boxes sizes have been roughly determined based on the ratio of defect to image in the LR dataset. In an HR image, the defect occupy a much smaller portion of the image. After the HR image is scaled to LR image size by Method A, any defects become even smaller making the detection outcomes imprecise. In Method B, most defects are detected. However, many false alarms (example region 660) are introduced because the method densely scans all sub regions of a HR image. Method C misses several scratches or cracks or identifies them imprecisely. The disclosed methods provide more accurate and precise defect detection outcomes as illustrated in the final row (labelled 'Our method') of the table below.

	P	R	F1
Method A	0.826	0.811	0.818
Method B	0.471	**0.988**	0.638
Method C	0.983	0.612	0.754
Our method	**0.998**	0.955	**0.976**

Table 1: Benchmarking results on crack dataset

	P	R	F1
Method A	0.940	0.746	0.832
Method B	0.483	**0.990**	0.649
Method C	0.948	0.113	0.202
Our method	**0.979**	0.951	**0.965**

Table 2: Benchmarking results on scratch dataset

[0060] The reference in this specification to any prior publication (or information derived from it), or to any matter which is known, is not, and should not be taken as an acknowledgment or admission or any form of suggestion that that prior publication (or

information derived from it) or known matter forms part of the common general knowledge in the field of endeavor to which this specification relates.

[0061] Throughout this specification and the claims which follow, unless the context requires otherwise, the word "comprise", and variations such as "comprises" and "comprising", will be understood to imply the inclusion of a stated integer or step or group of integers or steps but not the exclusion of any other integer or step or group of integers or steps.

[0062] The scope of this disclosure encompasses all changes, substitutions, variations, alterations, and modifications to the example embodiments described or illustrated herein that a person having ordinary skill in the art would comprehend. The scope of this disclosure is not limited to the example embodiments described or illustrated herein. Moreover, although this disclosure describes and illustrates respective embodiments herein as including particular components, elements, feature, functions, operations, or steps, any of these embodiments may include any combination or permutation of any of the components, elements, features, functions, operations, or steps described or illustrated anywhere herein that a person having ordinary skill in the art would comprehend. Although this disclosure describes or illustrates particular embodiments as providing particular advantages, particular embodiments may provide none, some, or all of these advantages.

Claims

1. A system for defect detection in images, the system comprising:

   one or more processor(s);

   a memory comprising instructions executable by the processor(s) to:

   receive a target image for defect detection;

   process the target image using a reinforcement learning agent (RL agent) to identify a plurality of coarse defect regions in the target image;

   process each coarse defect region using an object detection model (OD model) to identify a sub-region of the coarse defect region, the sub-region corresponding to a defect;

   wherein the OD model is trained using a low-resolution image dataset; and

   the RL agent is trained using a high-resolution image dataset and a dynamic threshold setting mechanism for the RL agent's reward function.

2. The system of claim 1, wherein processing the target image using the RL agent to identify the coarse defect region comprises: region selection operation; and region refinement operation.
3. The system of claim 2, wherein the RL agent is trained to optimize losses between the region selection operation and the region refinement operation.
4. The system of claim 2, wherein the RL agent is trained to perform coarse defect region selection operation based on feedback from the OD model.
5. The system of claim 3 or 4, wherein the RL agent is trained in a first phase to minimize losses for the defect region selection operation.
6. The system of claim 5, wherein the RL agent is trained in a second phase to minimize losses for both defect region selection operation and region refinement operations.

   .

7. The system of any one of claims 1 to 6, wherein the RL agent is trained using the reward function $R(s,a) = \begin{cases} 1 + IOU_{diff} & if\ IOU_{det} \geq IOU_{th} \\ 0 & else \end{cases}$

   wherein IOU_{det} is the Intersection Over Union(IOU) between ground truth defect and detected defect,

   IOU_{diff} is the difference of IOU_{det} before and after region refinement, and

   IOU_{th} is the threshold.

8. The system of any one of claims 1 to 7, wherein the defect comprises a crack or a scratch on a surface.

9. The system of any one of claims 1 to 8, wherein the high-resolution image dataset is a dataset of high-resolution dataset of unannotated images.

10. A method for defect detection in images, the method comprising:

    receiving, at a processor, a target image for defect detection;

    processing the target image using a reinforcement learning agent (RL agent) provided in a memory accessible to the processor to identify a coarse defect region in the target image;

    processing each coarse defect region using an object detection model (OD model) provided in the memory to identify a sub-region of the coarse defect region corresponding to a defect;

    wherein the OD model is trained using a low-resolution image dataset; and

    the RL agent is trained using a high-resolution image dataset and a dynamic threshold setting mechanism for the RL agent's reward function.

11. The method of claim 10, wherein processing the target image using the RL agent to identify the coarse defect region comprises: region selection operation; and region refinement operation.

12. The method of claim 11, wherein the RL agent is trained to optimize losses between the region selection operation and the region refinement operation.

13. The method of claim 11, wherein the RL agent is trained to perform coarse defect region selection operation based on feedback from the OD model.
14. The method of claim 12 or 13, wherein the RL agent is trained in a first phase to minimize losses for the defect region selection operation.
15. The method of claim 14, wherein the RL agent is trained in a second phase to minimize losses for both defect region selection operation and region refinement operations.
16. The method of any one of claims 10 to 15, wherein the RL agent is trained using the reward function $R(s,a) = \begin{cases} 1 + IOU_{diff} & if\ IOU_{det} \geq IOU_{th} \\ 0 & else \end{cases}$

    wherein IOU_{det} is the Intersection Over Union(IOU) between ground truth defect and detected defect,

    IOU_{diff} is the difference of IOU_{det} before and after region refinement, and

    IOU_{th} is the threshold.
17. The method of any one of claims 10 to 16, wherein the defect comprises a crack or a scratch on a surface.
18. The method of any one of claims 10 to 17, wherein the high-resolution image dataset is a dataset of high-resolution dataset of unannotated images.