Please start by saving this file with the name: GUID_Surname_UESTC4006P_Preliminary_year
******* Please add appropriate course code***

Student Name	Bingyi Liu	Placement Company (if appropriate)	
Student Matriculation Number	2839982L	Working Title of Project	Design and Implementation of Road Crack Detection System Based on YOLO Network Model
UESTC Student Number	2022190904030	Name of First Supervisor	Yulin Wang
Degree programme	BEng (Hons) Electronics and Electrical Engineering	**Declaration of Originality and Submission Information**	*I affirm that this submission is all my own work in accordance with the University of Glasgow and UESTC Regulations, and the James Watt School of Engineering requirements.* Signed (Student): Bingyi Liu
Academic year	2025 - 2026		

Your report should be NO more than 8 pages in length and include the below subject headings and incorporated within this document:

Project Description (no more than half a page)
Measurable Outcomes (no more than half a page): including Main Tasks and Targets, and Tangible outcomes (Hardware, Software, Hardware & Software, Theoretical research)
Technical Background (at least four pages)**:** including Literature Review, Topic Basis and Significance, and Research Status and Development Trend
References
Work Plan (no more than one page): including Project Outline

Resources: Complete the component request form and email the form to your 1st supervisor separately.
Risk Assessment Form: You must submit a Risk Assessment Form. Please have it signed with your 1st supervisor.

Deadlines for submission of this report: Please upload this report via the Moodle page and GC-UESTC FYP management system by the deadline mentioned in Table 1 of your project handbook.

Project Specification & Preliminary Report

1. Project Description

Road surface cracks are one of the tell-tale signs of road deterioration and a key factor affecting traffic safety, road water ingress and settlement, as well as maintenance costs. Consequently, detecting road surface cracks is paramount in road maintenance work. However, traditional manual surveys are time-consuming and labour-intensive, with detection results prone to significant error. Moreover, working conditions on certain sections are hazardous, potentially posing a threat to personnel safety. Employing the YOLO Network Model enables standardised processing of detection results while reducing operational costs for road maintenance departments.

This project will utilise publicly available datasets for training, alongside constructing a small validation set comprising approximately 100 to 300 images to assess the model's generalisation capabilities. The final deliverable will be an executable programme supporting both image and video detection. This will accommodate batch image folders, offline video files, and potential camera inputs. A graphical user interface developed using Qt will provide swift inference, visualisation, and result export functionality.

The project is anticipated to run on an AMD Ryzen 9 5900HX + NVIDIA RTX 3080 Laptop GPU platform. Training scripts, prediction scripts, and their reproducible configurations will be provided alongside the executable to ensure project traceability and engineering quality.

2. Measurable Outcomes

2.1. Main Tasks

- **Model Development:** Compile a dataset of road surface cracks and create a validation set comprising approximately 100 to 300 images; train models using YOLOv8 and YOLOv11 architectures and compare their performance.

- **Deliverables:** Package the inference files into a Windows executable built with Qt, enabling users to process images, image folders, or video files, including real-time processing of camera footage.

- **Supplementary Files:** Export ONNX or TensorRT engines, provide reproducible training and inference scripts, and compile a concise user manual detailing project usage.

2.2. Targets

- **Image Detection:** Support single-image detection and batch processing of folder contents; visualise results and export detections in CSV/JSOM/COCO formats.

- **Video Detection Functionality:** Supports processing local videos and, where feasible, real-time camera feeds; visualises results and permits exporting detection outputs.

- **Local Execution Capability:** This project will operate on an AMD Ryzen 9 5900HX + NVIDIA RTX 3080 Laptop GPU platform, featuring a GUI interface for adjusting model parameters, detection thresholds, and output paths.

- **Reproducibility:** This project will provide reproducible training and inference scripts alongside other files to ensure consistent implementation across different devices.

2.3. Tangible Outcomes

- **Hardware:** AMD Ryzen 9 5900HX + NVIDIA RTX 3080 Laptop GPU platform, with optional USB camera for external dataset storage.

- **Software:** Python and PyTorch for training code; Ultralytics for configuring YOLOv8 and YOLOv11 models; ONNX or TensorRT for inference; Qt6 for GUI development; alongside data preprocessing scripts, model scripts with varied weights, and executable programmes.

- **Hardware & Software:** A standalone executable file containing various configuration options and operational examples; fundamental usage metrics on target platforms, including memory utilisation and qualitative latency.

- **Theoretical Research:** Concise literature review on pavement crack detection; concise literature review on YOLO Network Models; principles of model attention mechanisms or lightweight enhancements; analysis plan for potential issues and brief explanation of dataset origins.

3. Technical Background

3.1. Literature Review

Research into pavement crack detection has followed a clear trajectory over the past decade: progressing from traditional visual detection methods centred on image thresholding, edge characteristics and morphological features, to machine learning frameworks employing texture or shape features alongside classifiers. This evolved further into the use of convolutional neural networks for semantic segmentation and object detection. More recently, global representation approaches incorporating Transformers and end-to-end detectors have been introduced.

Without exception, these developments have addressed the challenges inherent in detecting road cracks: their slender form, low contrast, and strong background interference, while simultaneously meeting operational requirements across diverse equipment, materials, and lighting conditions.

In early work, *CrackTree* treated road cracks as growable curve structures. To handle complex lighting and coarse textures in real-world conditions, it first suppressed textures and removed shadows, then enhanced linear consistency using the Zhang Liang voting algorithm. Finally, it employed minimum spanning tree or minimum cost path methods to track continuous skeleton structures, effectively circumventing the common breakage and missed detection issues inherent in traditional threshold-based and edge-feature-dependent approaches [1].

CrackIT further advanced this by systematically constructing a complete workflow from preprocessing and candidate extraction through connection and skeletonization to geometric quantification. This established a comprehensive research framework enabling reproducible experiments and unified comparisons, proposing a procedural paradigm of ‘detection-characterisation-quantification’ [2]. The two representative studies described above have established geometric constraints and topological consistency as pivotal directions in crack image processing research, providing stable reference samples for subsequent diverse processing methodologies.

However, as the scenarios for detecting road surface cracks grow increasingly complex, traditional methods become increasingly fragile and inadequate for such intricate tasks. Against this backdrop, researchers have begun introducing data-driven learning paradigms.

CrackForest/Random Structured Forests formulates crack modelling as a discriminative problem involving structured fragments. By leveraging the random forest architecture, it learns to distinguish between ‘linear structures’ and ‘background textures’ at the patch level. The development process also disclosed corresponding urban road imagery and evaluation protocols, significantly advancing synergistic progress in both data and research methodologies [3].

Concurrently, *SDNET2018* compiled over 56,000 concrete crack and non-crack images, covering most usage scenarios from bridge decks and walls to road surfaces. It retained common background disturbances such as shadows, stains, and holes, substantially expanding supervised samples to become one of the most widely used datasets in deep learning [4].

The *CRACK500* dataset, focused on asphalt pavements, provides high-resolution, pixel-level annotated samples. This offers a more challenging testing platform for fine-grained semantic segmentation, encouraging greater attention to ‘fine crack continuity and small object recall’ in network architectures and facilitating comparative research [5].

These distinct datasets, each emphasising different research directions, have enabled comparable and reproducible comparative studies across diverse methodologies and target domains.

As research focuses more and more on deep learning, the advantages of convolutional networks in the field of crack detection have gradually become more prominent, especially in terms of multi-scale fusion and deep supervision strategies.

For example, the *DeepCrack* model significantly improves the coherence of slender cracks and the integrity of crack boundaries by hierarchically aggregating shallow textures and high-level semantics in the decoding stage and introducing side outputs and supervision signals at multiple scales [6].

The *FPHBN* model further innovates on this basis, combining feature pyramids with hierarchical hard example reweighting, combining the "top-down + bottom-up" fusion strategy and sample redistribution method, effectively solving the two key problems of "weak contrast + strong interference" and showing more stable consistency on multiple datasets [7].

However, in actual engineering applications, inference latency is another problem that cannot be ignored. To address this problem, *FPCNet* introduces multi-expanded convolution and SE attention mechanism in the encoding-decoding architecture to achieve lightweight feature fusion. The above improvements not only ensure pixel-level accuracy but also effectively reduce computational costs [8].

At the same time, other studies systematically compared the training paradigms of *U-Net* and its improved versions on datasets such as *CFD* and *CRACK500*, including strategies such as encoder transfer learning, one-cycle learning rate, and strong data augmentation. These studies provide clear implementation methods for the "small data + strong generalization" training route [9]. The above work has established the status of pixel-level segmentation in scenarios requiring geometric quantization and continuity.

On the other hand, in recent years, target detection methods oriented towards real-time performance and deployment links have developed rapidly and gradually matured. Researchers usually regard cracks as "regions" or "instances" that need to be reviewed, prioritizing rapid coverage of large areas and long distances, and then refining the candidate regions.

A large number of improvements based on the *YOLO* family are representative of this trend. For example, on the *YOLOv8m* framework, the recall rate of very fine cracks is significantly improved through decoupling heads, feature fusion, and loss reconstruction, while reducing the number of parameters. The above scheme has achieved stable results in the public road crack dataset [10].

The *MS-YOLOv8* improvement uses multi-scale attention and structural rearrangement to achieve a balance between speed and accuracy, effectively alleviating the natural disadvantage of "weak contrast + small targets" mentioned above [11].

EMG-YOLO takes a different approach and significantly improves the robustness of the model when deployed on edge devices from the perspective of "multi-granularity representation and morphological perception" [12].

YOLOv7-WMF, on the other hand, focuses on introducing instance segmentation capabilities into the detection framework, enabling the output to directly include mask-level contour information, facilitating subsequent geometric calculations of width and direction [13].

The aforementioned extensive improvements based on the YOLO family have shown that in the actual "survey-review-disposal" workflow, the detection-first two-stage system often has lower annotation costs, a more complete export and deployment chain, and more controllable online latency. This system not only improves work efficiency but also reduces overall costs, providing strong support for practical engineering applications.

Nowadays, with the continuous development of visual *Transformer* technology, long-range dependencies and global context information have gradually been introduced into the field of crack detection, which has greatly enriched the research on the attention mechanism of crack detection.

For example, *SwinCrack* realizes cross-window information fusion through a hierarchical self-attention mechanism, which can obtain more continuous and detailed crack expressions under non-uniform lighting and weak contrast conditions [14]. *iSwin-UNet* maintains the ability of pixel-level segmentation through a hierarchical window attention mechanism under a lightweight structure [15].

In addition, some studies have introduced cross-shaped or strip-shaped attention mechanisms in the encoder to strengthen the directional representation, so that the network can more effectively distinguish slender linear structures such as cracks, road seams and markings that are similar in morphology but different in semantics [16].

In terms of object detection, the *RT-DETR* family has proposed a series of innovative strategies, including efficient hybrid encoders, query selection and end-to-end training, making "no NMS required, real-time" end-to-end detection possible, and has demonstrated comparable performance to the advanced *YOLO* family on small objects and complex background tasks [17].

The subsequent *RT-DETRv3* further accelerated the convergence speed and improved the recall rate through dense positive sample level supervision, providing a new engineering baseline for targets with “fine, sparse, and easily confused” features such as cracks [18].

In addition, in response to the problem of noise labelling that is difficult to avoid in real scenes, recent studies have begun to explicitly model the distribution of label noise and adopt robust loss and adaptive reweighting strategies to alleviate boundary jitter and overfitting [19]. In addition, at the methodological level, some literatures have summarized the key factors affecting crack detection and segmentation performance, such as resolution, imaging angle, lighting and shadow, material, and mixed interference, and also discussed the corresponding methodological trends, including multi-scale feature aggregation, attention mechanism, weak/semi-supervised learning, and cross-domain adaptation, providing a systematic reference for selection and evaluation [20], [21].

In terms of enhancement modules, lightweight self-attention mechanisms such as *CBAM* and coordinate attention have been used as pluggable units in many papers and are often used to improve the sensitivity of extremely fine textures without significantly increasing latency [22], [23].

It is worth noting that the 2D texture method is still difficult to completely eliminate ambiguity under certain extreme conditions. Therefore, some studies have directly used 3D contours or line laser point clouds to extract cracks in geometric space, thereby improving the reliability of width and depth estimation [24].

3.2. Topic Basis and Significance

From a whole-life road asset management perspective, cracks are not only among the earliest visible distresses; they're the thin end of the wedge, setting off the chain of failures that follows—alligator cracking, potholes, and subsidence. Crack formation and propagation are driven by a variety of factors, including load, temperature and humidity, ultraviolet light, and material aging. Once cracks appear, they significantly increase the risk of pavement water seepage and freeze-thaw cycles, accelerating structural damage, leading to a nonlinear increase in safety hazards and maintenance costs. Traditional maintenance methods, relying on manual inspections and subjective judgment, are no longer able to meet today's network-wide, regularized, and objectively consistent maintenance needs.

Automated inspection technology, cantered around computer vision, offers a new solution. First, it upgrades "sampling" to "full-volume" inspection, achieving a spatial-temporal profile of the damage through mileage-based statistics and time series tracking. Second, it converts detection and segmentation results into structured evidence (such as location, mask, or approximate width/length, direction, and connectivity), which is directly integrated with the work order system, forming a closed loop of "discovery-assessment-action-retrospection." Finally, it introduces real-time and reproducibility to vehicle-mounted, airborne, or backpack platforms, allowing different work teams to obtain comparable quantitative conclusions at different times [1], [4]– [7].

From an academic perspective, crack detection has the characteristics of small targets, weak contrast, tree topology, and strong domain shift, making it an ideal scenario for testing multi-scale features, attention mechanisms, end-to-end detection, noise robust learning, and cross-domain adaptation capabilities. From an engineering perspective, the mature export of *YOLO* and *RT-DETR* (such as *ONNX/TensorRT*) as well as sparse attention and lightweight backbone networks make the "last mile" of "offline GUI tools + edge deployment" a replicable solution [10]– [13], [17], [18].

Therefore, this project not only focuses on the core demands of "efficiency-consistency-auditability" in real road maintenance but also helps to precipitate generalizable algorithms and engineering experience on typical difficulties (such as weak contrast, strong interference, noise labelling, and domain shift).

3.3. Research Status

At the current stage, semantic segmentation and object detection each have their own advantages in the task of pavement crack detection and are developing in parallel. When the task requires accurate geometric quantities (such as width, length, branching degree) and topological continuity, multi-scale decoding, side-output deep supervision, and hard case reweighting technologies represented by *DeepCrack*, *FPHBN*, and *FPCNet* are still reliable choices [6]– [9]. At the same time, under complex and non-uniform lighting conditions, the introduction of global attention mechanisms such as *Swin* or *CSWin* can further reduce the false detection rate and repair the broken parts [14]– [16].

On the other hand, when the scene emphasizes throughput, low latency, and deployment chain (such as long-distance vehicle census or offline batch processing), the detection-first paradigm has gradually become the mainstream choice due to its low annotation cost (box/point), mature deployment ecology, and considerable real-time performance [10]– [13]. On this basis, by superimposing lightweight segmentation or refinement operations on the candidate area to obtain geometric quantities, a more reasonable balance can be achieved between "accuracy-cost-engineering feasibility".

In addition, a consensus has gradually been formed around the practice of "stability and generalization". At the data level, using "shadows, reflections, stains, markings, wet surfaces after rain" as templates, we design targeted data augmentation and hard-case sampling strategies; at the evaluation level, we conduct cross-dataset validation and establish a small and precise local validation set; at the learning level, we improve the model's robustness to domain shift and noisy labels through attention mechanisms, contrastive learning, and weak/semi-supervised learning [19]–[21], [23].

3.4. Development Trend

In the future, the intersection of research and application may further tilt towards the "evaluation and decision-making" level. First, the integration of detection, segmentation and quantification will become the norm. The output results will no longer be limited to "presence/absence of cracks" but will be expanded to include structured indicators such as "width, length, direction, connectivity and density", and will be bound to the normalized rating model to directly serve the work order priority sorting and resource allocation.

Secondly, the end-to-end real-time detection ecosystem will become more mature. The *RT-DETR* family has demonstrated the feasibility of "no NMS + high FPS". Combined with lightweight convolutional backbone and sparse attention mechanism, it is expected to further improve the recall rate in small target scenarios such as cracks, while reducing the complexity of post-processing [17], [18].

Thirdly, lightweight attention mechanism and model distillation, pruning, and quantization may become the standard for edge deployment. This not only meets the computing power budget of laptops or embedded devices, but also maintains sensitivity to extremely fine textures [11], [12], [22], [23].

It is worth noting that weak/semi-supervised learning and synthetic data may also effectively alleviate the bottleneck of high-cost pixel annotation. In the setting of noisy labels and incomplete annotations, boundary prediction can be stabilized through self-training and uncertainty modelling [19]– [21].

In addition, 2D/3D fusion technology will be more widely used in high-risk scenarios such as airport runways and bridge decks. Using 3D contours or point cloud data to correct the ambiguity of 2D textures can significantly improve the accuracy of width estimation and enhance the interpretability of "structural cracks vs. surface texture" [24].

Finally, in the field of engineering processes, *MLOps* centred on "data closure-traceability-comparability" will gradually sink to the front line of inspection. From data collection, annotation, training, evaluation to export, deployment and GUI interaction, standardized configuration, log and version management are formed, so that "cross-team and cross-cycle" comparisons are not only statistically significant but also legally effective.

The above trends are closely related to the actual workflow and also provide a clear direction for the system implementation and subsequent expansion of this project.

4. References

[1] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, “CrackTree: Automatic crack detection from pavement images,” Pattern Recognition Letters, vol. 33, no. 3, pp. 227–238, 2012. doi: 10.1016/j.patrec.2011.11.004.

[2] H. Oliveira and P. L. Correia, “CrackIT—An image processing toolbox for crack detection and characterization,” in Proc. IEEE Int. Conf. Image Process. (ICIP), 2014.

[3] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, “Automatic road crack detection using random structured forests,” IEEE Trans. Intelligent Transportation Systems, vol. 17, no. 12, pp. 3434–3445, 2016. doi: 10.1109/TITS.2016.2552248.

[4] M. Maguire, S. Dorafshan, and R. J. Thomas, “SDNET2018: A concrete crack image dataset for machine learning applications,” Dataset, Utah State University, 2018. doi: 10.15142/T3TD19.

[5] M. Sabouri et al., “SUT-Crack: A comprehensive dataset for pavement crack detection under varied conditions,” Data in Brief, vol. 48, 109193, 2023.

[6] Q. Zou et al., “DeepCrack: Learning hierarchical convolutional features for crack detection,” IEEE Trans. Image Processing, vol. 28, no. 3, pp. 1498–1512, 2019. doi: 10.1109/TIP.2018.2878966.

[7] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling, “Feature Pyramid and Hierarchical Boosting Network for pavement crack detection,” IEEE Trans. Intelligent Transportation Systems, vol. 21, no. 4, pp. 1525–1535, 2019.
[8] W. Liu, Y. Huang, Y. Li, and Q. Chen, “FPCNet: Fast pavement crack detection network based on encoder–decoder architecture,” arXiv:1907.02248, 2019.
[9] S. L. H. Lau, E. K. P. Chong, X. Yang, and X. Wang, “Automated pavement crack segmentation using U-Net-based CNN,” arXiv:2001.01912, 2020.
[10] R. Zeng, Z. He, and W. Li, “A lightweight pavement crack detection method based on improved YOLOv8m,” Scientific Reports, vol. 14, 16297, 2024.
[11] C. Han, H. Tian, C. Li, and Q. Wang, “MS-YOLOv8: Pavement crack detection using multi-scale feature enhancement,” Sensors, vol. 24, no. 7, 2024.
[12] C. Xing, Y. He, S. Zhou, X. Zhang, and Z. Ma, “EMG-YOLO: Enhanced multi-granularity YOLO for pavement crack detection,” Frontiers in Neurorobotics, vol. 18, 2024.
[13] C. Ye, M. Joseph, M. Haldar, and C. C. Caragea, “YOLOv7-WMF: A lightweight YOLOv7 with mask-guided fusion for concrete crack instance segmentation,” Automation in Construction, vol. 159, 105393, 2024.
[14] L. Wang et al., “Asphalt pavement crack detection based on Swin-Transformer,” Digital Signal Processing, vol. 139, 104061, 2023.
[15] Q. T. Le and T. T. Nguyen, “iSwin-UNet for pavement crack segmentation,” Buildings, vol. 14, no. 6, 2024.
[16] Z. Chen, H. Bian, W. Chen, and J. Xu, “A crack segmentation strategy using CSW-S network with cross-shaped window attention,” Scientific Reports, vol. 14, 13539, 2024.
[17] Z. Chen et al., “RT-DETR: Real-Time DEtection TRansformer,” in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2024. (arXiv:2304.08069)
[18] S. Liu et al., “RT-DETRv3: Efficient real-time detection with hybrid encoder and dense supervision,” in Proc. IEEE/CVF Winter Conf. Applications of Computer Vision (WACV), 2025.
[19] X. Li et al., “Distribution-aware noisy-label crack segmentation,” arXiv:2410.09409, 2024.
[20] H. Gong, L. Liu, H. Liang, Y. Zhou, and L. Cong, “A state-of-the-art survey of deep learning models for automated pavement crack segmentation,” International Journal of Transportation Science and Technology, vol. 13, pp. 44–57, 2024.
[21] Q. Yuan, X. Zhang, X. Li, and D. Zhang, “A review of computer vision-based crack detection methods,” Remote Sensing, vol. 16, no. 16, 2910, 2024.
[22] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, “CBAM: Convolutional block attention module,” in Proc. ECCV, 2018, pp. 3–19.
[23] Q. Hou, D. Zhou, and J. Feng, “Coordinate attention for efficient mobile network design,” in Proc. CVPR, 2021, pp. 13713–13722.
[24] Z. Pan, X. Cao, J. Ng, and W. Chai, “One-stage 3D profile-based pavement crack detection and quantification,” Automation in Construction, vol. 148, 104735, 2023.

5. Work Plan

5.1. Project Outline

Develop an offline road-crack detection system, compare YOLOv8 and YOLOv11, integrate small-target enhancements, validate via ablation studies, export to ONNX/TensorRT, and package as a Qt desktop application. Ensure reproducibility with fixed random seeds throughout the project.

5.2. Timeline

- **Nov–Dec 2025:** Define scope, organize data, build local set, set up environment.
- **Jan–Feb 2026:** Train/evaluate YOLOv8/11, record metrics, identify errors.
- **Mar 2026:** Add lightweight attention (e.g., CBAM), adjust small-target policies, conduct ablations.
- **Early Apr 2026:** Export to ONNX/TensorRT, benchmark latency, finalize parameters.

- **Mid–Late Apr 2026:** Develop Qt6 app, cross-dataset testing, package executable, freeze code, submit manual, prepare final report and presentation.

5.3. Quality Gates

- **Local Set:** mAP@.5 ⩾ 0.60, recall ⩾ 0.80.
- **Performance:** ⩾ 20 FPS at 1080p in GUI.
- **Reproducibility:** One-command run, fixed seeds, traceable configs.

5.4. Deliverables

- **Models:** YOLOv8/11 models, ONNX/TensorRT engines.
- **Application:** Qt desktop app supporting images/folders/videos with visualization and structured export.
- **Documentation:** User manual, logs, scripts, final report.

6. Resources

This project will be primarily completed offline on a personal laptop (Ryzen 9 5900HX + RTX 3080, ⩾32 GB of RAM, ⩾1 TB of SSD). Supporting resources include a USB/mobile phone camera for data acquisition and an external SSD for backup. The software environment includes Windows 11/Ubuntu (or WSL2), Python 3.x, PyTorch 2.x (CUDA 12.x), Ultralytics YOLOv8/v11, ONNX/TensorRT, Qt 6, and OpenCV. Public datasets (CFD, CRACK500, SDNET2018, SUT-Crack) will be used, along with a self-built local validation set of 100–300 images. LabelImg/Roboflow will be used for annotation, and Git/DVC will be used for version control. A fixed random seed will be used to ensure reproducibility. Data and open-source licenses will be strictly adhered to, and the project will be used solely for course research and demonstration purposes. Any additional hardware will be approved according to the school's procedures.