Journal of Research in Science and Engineering (JRSE)
ISSN: 1656-1996 Volume-7, Issue-11, November 2025
DOI: 10.53469/jrse.2025.07(11).04

Design and Implementation of a Ship Target Detection System Based on PyQt

Wanqiu Xu, Jialu Sun, Binghe Zhang

Jiangsu Maritime Institute, Nanjing 211170, Jiangsu

Abstract: ***With the advancement of intelligent shipping and smart port construction, ship target detection plays a crucial role in maritime traffic safety, port monitoring, and environmental protection. Aiming at the shortcomings of traditional monitoring systems in real-time performance and visualization, this paper designs and implements a ship target detection system based on the PyQt framework. The system utilizes a deep learning model to detect ship targets in videos and employs PyQt to construct a visual interactive interface, enabling real-time display and management of detection results. The system implements core functions such as data management, detection processing, and result presentation, and exhibits good real-time performance and stability. It provides a technical foundation for subsequent multi-source data fusion and the expansion of intelligent monitoring systems.***

Keywords: PyQt, Ship Target Detection, Deep Learning, Visualization System**.**

1. Introduction

In recent years, the rapid development of the global shipping industry has led to a continuous increase in maritime traffic density. Ship monitoring and target detection have become increasingly critical in maritime supervision, port scheduling, and marine safety protection. Traditional monitoring methods mainly rely on radar detection and manual video observation, which not only suffer from limitations such as high workload intensity and low recognition accuracy but also struggle to achieve large-scale and all-weather automated monitoring. To enhance the intelligence level of port and maritime traffic management, ship detection technology based on computer vision and deep learning has become a core research direction at present.

In terms of technical research, relevant scholars have conducted multi-dimensional explorations. Huang Jie et al. [1] proposed a ship target detection method combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM); Zhang Xu [2] designed a ship detection scheme based on template matching and deep learning for port scenarios; Sheng Mingwei et al. [3] focused on addressing the problems of complex detection scenarios and insufficient accuracy by optimizing the YOLOv3 algorithm—through dataset augmentation using Mixup, introducing an attention mechanism into the Darknet-53 network, and constructing a saliency detection branch for secondary optimization, which effectively reduced the missed detection rate. In terms of dataset construction, Shao et al. [4] screened and built the inland ship dataset "SeaShips" in 2018, covering 6 common types of ships and demonstrating high practicality; for this dataset, their team further proposed a saliency-aware detection method [5], which uses CNN to predict the type and position of ships, corrects the position with global contrast saliency region detection technology, and integrates coastline data to achieve high-precision classification and detection. In addition, Guang Ruizhi et al. [6] proposed the FoveaSDet algorithm for UAV aerial images of ships in waterways. By adopting a single-stage anchor-free detection architecture and Foveahead to handle scale variations, the algorithm significantly improved the detection accuracy of small targets.

In terms of development tools, the rapid development of the Python ecosystem provides support for technology implementation. As a fully functional graphical interface development framework, PyQt [7] features rich components, a flexible signal-slot mechanism, and cross-platform capabilities. It can seamlessly integrate with deep learning frameworks such as PyTorch and TensorFlow, laying an ideal technical foundation for building an integrated and interactive ship detection system.

Based on the aforementioned research and technical conditions, this paper designs and implements a visual ship target detection system using PyQt as the development framework and combining it with the YOLOv8 deep learning target detection algorithm. The system covers the complete process from image/video input and target detection to result visualization, enabling intuitive presentation of detection results and supporting interactive operations.

2. System Design

2.1 Feasibility Analysis

2.1.1 Technical Feasibility

From a technical perspective, the technologies required for system development are relatively mature, demonstrating good feasibility:

1) PyQt Framework: PyQt is a Python binding based on the Qt library, featuring cross-platform compatibility, rich functionality, and ease of development. Currently, PyQt has been widely used in graphical user interface (GUI) development, with comprehensive documentation and abundant open-source resources. Developers can quickly master its usage and efficiently develop high-quality interface applications.

2) Target Detection Algorithms: Target detection algorithms in the field of deep learning have achieved significant progress. YOLO series algorithms (e.g., YOLOv5, YOLOv8) offer advantages of fast detection speed and high accuracy, which can meet the requirements of real-time ship detection. Meanwhile, these algorithms are highly open-source,

allowing developers to conduct secondary development based on open-source code and reducing the difficulty of algorithm integration.

3) Image Processing Technology: The OpenCV library in Python provides a wealth of image processing functions, such as image reading, scaling, filtering, and edge detection. These functions can meet the system's needs for ship image preprocessing and improve the accuracy of subsequent target detection.

4) Development Tools and Environment: The system can be developed using PyCharm as the integrated development environment (IDE). This tool supports Python code writing, debugging, and execution, providing a good development experience. Additionally, the Python language is characterized by concise syntax and easy maintenance, which can improve the efficiency of system development.

2.1.2 Economic Feasibility

The economic cost of system development is relatively low, indicating high economic feasibility:

1) Software Costs: All software required for system development is open-source or free, such as Python, PyQt, OpenCV, and open-source code of YOLO algorithms. No software copyright fees need to be paid, significantly reducing development costs.

2) Hardware Costs: The system has moderate requirements for hardware configuration. Ordinary desktop computers or laptops can meet the needs of development and operation, eliminating the need to purchase expensive dedicated hardware equipment.

3) Maintenance Costs: The system adopts a modular design with a clear code structure, facilitating subsequent maintenance and upgrades. Meanwhile, development based on open-source technologies enables access to extensive community support, reducing maintenance difficulty and costs.

2.1.3 Operational Feasibility

The system emphasizes user operation experience and demonstrates good feasibility in terms of operation:

1) Interface Design: A intuitive and user-friendly GUI is developed using PyQt, with a reasonable interface layout and clear function buttons. Users do not require professional computer knowledge and can proficiently operate the system after simple training.

2) Operation Process: The system's operation process is concise and clear. Users can complete ship detection tasks through just a few simple operations (e.g., loading images, clicking the detection button, and viewing results), avoiding complex operation steps.

3) Error Handling: The system is equipped with a comprehensive error handling mechanism. When users perform improper operations, the system will pop up prompt messages to guide correct operations, reducing the impact of user operation errors.

2.2 System Process Design

The overall workflow of the system is shown in Figure 1, which mainly includes three core stages: data input stage, target detection stage, and result presentation stage. The specific process is as follows:

1) Data Input Stage: The system supports importing images or video streams from local files and can also select real-time camera video as the input source. This stage completes data reading and format standardization processing to ensure efficient operation of subsequent algorithms.

2) Target Detection Stage: The core of the system adopts the YOLOv8-based ship detection algorithm [8] to perform feature extraction and target recognition on input images, and outputs information such as bounding boxes and confidence levels of ship targets. The algorithm model is loaded during system initialization, and the detection process is performed in an independent thread to ensure smooth interface response.

3) Result Presentation Stage: The PyQt front-end interface displays detection results in real-time and performs visual annotations on detected targets, including rectangular boxes, category labels, and confidence levels. Users can choose to save detection images or export result files, and the system supports automatic generation and storage of detection records.

Figure 1: System Process Design

2.3 Functional Module Design

The system's functional modules mainly include: data management module, detection processing module, and result presentation module. The functions of each module are shown in Table 1:

Table 1: Functional Module Design

Module Name	Function Description
Data Management Module	Responsible for reading images and videos
Detection Processing Module	Performing ship target recognition using the YOLOv8 model
Result Presentation Module	Displaying detection results in the PyQt interface

3. System Implementation

3.1 Development Environment and Technical Framework

Table 2: Development Environment and Technical Framework

Item	Specification
Operating System	Windows 10
Development Language	Python 3.8
Main Dependent Libraries	opencv-python, PyQt5, Ultralytics, PyTorch
IDE Environment	PyCharm

The environment configuration required for system development and operation is shown in Table 2.

3.2 System Interface Design

From the perspective of functional design, the interface is divided into two core areas: the left area is a vertically arranged control component area, and the right area is a label component area responsible for displaying input and output content (see Figure 2). The control components are laid out from top to bottom according to the operation process, including: a single-line text box for displaying the model weight path, a button for selecting the model weight path, a drop-down box for selecting the detection mode, a button for selecting input images/videos, a start detection button, and a parameter setting button. Among them, clicking the parameter setting button will pop up a configuration panel, allowing adjustment of YOLOv8 inference parameters (consistent with the parameter types supported by the command line).

In terms of layout structure, the interface adopts a "three-vertical and one-horizontal" layout architecture: three vertical layouts are used to arrange control components, display input images, and show output results respectively, while a horizontal layout integrates the left and right areas to form the overall interface framework.

Figure 2: Interface Display

4. Conclusion

This paper designs and implements a ship target detection system based on the PyQt framework, integrating a deep learning model and a graphical visualization interface. The system can realize real-time detection and display of ship targets in port or maritime video streams. It operates stably, has a user-friendly interface, and features strong scalability, providing a reference and foundation for the development of subsequent intelligent maritime monitoring systems.

In the future, by integrating Automatic Identification System (AIS) data, the system will realize automatic matching between detection results and ship identity information, further improving its intelligence level and practical application value. Meanwhile, a Web front-end interface can be expanded to achieve remote monitoring and multi-source information sharing, providing more comprehensive technical support for smart ports and shipping safety.

Funding

1) Scientifc Research Project of Jiangsu Maritime Institute (2024ZKyb11);
2) 2024 Jiangsu Provincial College Students' Innovation and Entrepreneurship Training Program (GX-20240158).

References

[1] Huang J, Jiang Z G, Zhang H P, et al. Ship Target Detection in Remote Sensing Images Based on Convolutional Neural Network[J]. Journal of Beijing University of Aeronautics and Astronautics, 2017, 43(09): 1841-1848.
[2] Zhang X. Port Ship Detection and Recognition Method Based on Template Matching and Deep Learning[J]. Computer Application Technology, 2019, 4: 59-63.
[3] Sheng M W, Li J, Qin H D, et al. Ship Target Detection Algorithm Based on Improved YOLOv3[J]. Navigation and Control, 2021, 20(02): 95-109.
[4] Shao Z, Wu W, Wang Z, et al. SeaShips: A Large-Scale Precisely Annotated Dataset for Ship Detection[J]. IEEE Transactions on Multimedia, 2018, 20(10): 2593-2604.
[5] Shao Z, Wang L, Wang Z, et al. Saliency-Aware Convolution Neural Network for Ship Detection in Surveillance Video[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2019, 30(3): 781-794.
[6] Guang R Z, An B W, Pan S D. Ship Detection Algorithm for Aerial Waterway Images Based on Anchor-Free Network[J]. Computer Engineering and Applications, 2021, 57(15): 251-258.
[7] Rapid GUI Programming with Python and Qt; The Definitive Guide to PyQt Programming[J]. Scitech Book News, 2008, 32(1):
[8] Xu W. Research on Ship Target Detection Based on YOLOv8[J]. World Journal of Innovation and Modern Technology, 2025, 8(9): DOI: 10.53469/WJIMT.2025.08(09).14.