International Journal of Geo-Information

MDPI

Article

An Automated Method for Generating Prefabs of AR Map Point Symbols Based on Object Detection Model

Nixiao Zou, Qing Xu *, Yuqing Wu, Xinming Zhu and Youneng Su

Institute of Geospatial Information, Information Engineering University, Zhengzhou 450001, China; giser_znx_ieu@outlook.com (N.Z.); wyq2017ch@126.com (Y.W.); 13700875932@163.com (Y.S.)
* Correspondence: xq1982_no.1@163.com

Abstract: Augmented reality (AR) technology enables paper maps to dynamically express three-dimensional geographic information, realizing the fusion of virtual and real information. However, in the current mainstream AR development software, the virtual information usually consists of prefabricated components (prefabs), and the content creation for AR maps heavily relies on manual prefabrication. It leads to repetitive and error-prone prefabrication work, which restricts the design of the dynamic, interactive functions of AR maps. To solve this problem, this paper explored the possibility of automatically generating AR map prefabs using object detection models to establish a data conversion interface from paper maps to AR maps. First, we compared and analyzed various object detection models and selected YOLOv8x to recognize map point symbols. Then, we proposed a method to automatically generate AR map prefabs based on the predicted bounding boxes of the object detection model, which could generate prefabs with corresponding categories and positional information. Finally, we developed an AR map prototype system based on Android mobile devices. We designed an interaction method for information queries in the system to verify the effectiveness of the method proposed in this paper. The validation results indicate that our method can be practically applied to the AR map prefabrication process and can quickly generate AR map prefabs with high information accuracy. It alleviated the repetitive workload established through the manual prefabrication method and had specific feasibility and practicality. Moreover, it could provide solid data support for developing dynamic interactive functions of AR maps.

Keywords: AR map; object detection model; prefabs; point symbols; data conversion interface; AR development software

Citation: Zou, N.; Xu, Q.; Wu, Y.; Zhu, X.; Su, Y. An Automated Method for Generating Prefabs of AR Map Point Symbols Based on Object Detection Model. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 440. https://doi.org/10.3390 ijgi12110440

Academic Editors: Wolfgang Kainz, Peng Peng, Shu Wang, Maryam Lotfian, Feng Lu and Yunqiang Zhu

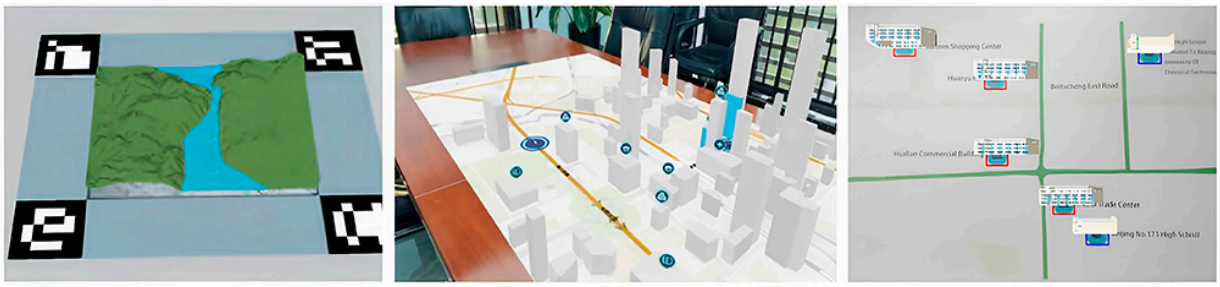
Received: 17 August 2023
Revised: 11 October 2023
Accepted: 20 October 2023
Published: 24 October 2023

1. Introduction

AR, a technology dedicated to merging the virtual and real worlds, has significant advantages in map representation and environmental integration. Augmented paper maps break the static visual representation of traditional maps and serve as an extension of traditional cartography and map visualization. AR can provide more immersive human–computer interaction and enhance the multidimensional dynamic perception of geographic information and understanding of the real world [1]. Therefore, an increasing number of cartographers are becoming involved in the research combining maps with AR [2–5], leading to the emergence of AR maps [6].

This paper takes paper maps as the research object and introduces AR technology to explore a new way of expressing geographic information data in order to expand its information load and enhance its practicality. Compared with traditional electronic maps, AR maps bring new vitality and possibilities to many ancient maps without digital forms, instead of monotonously assimilating them into electronic map forms. Typically, AR maps comprise real map environments and relevant virtual information [5]. Virtual information, often prefabricated information, is predesigned by humans and manifests explicitly as a series of prefabs within existing AR development software. These prefabs can range from

simple visual markers to complex interactive logic and are primarily used to create virtual objects rapidly and consistently within the AR environment. Existing research on AR maps mainly employs computer-vision-based methods for the global registration of prefabricated information and paper maps, thereby achieving high-quality fusion between the virtual and real elements [7–11], as shown in Figure 1. Implementation approaches mainly include 3D registration techniques based on natural features, marker-based 3D registration techniques, and hybrid tracking and registration techniques [12–14]. Ann Morrison et al. [15] developed the first mobile AR map system, Maplens, which utilized natural features of maps in conjunction with smartphones. Xu et al. [9] proposed an adaptive approach using a map symbol library to process symbol information. By combining the natural features of paper maps with artificially placed markers distributed in the corners, they achieved the local and global augmentation of terrain information. The Yousuya laboratory at the University of Southern California significantly improved the accuracy of AR systems by combining computer vision methods with inertial sensors [16]. These studies have made significant explorations and analyses in integrating AR technology with maps. However, these approaches often involve experimental and independent development solutions that focus on achieving the fusion of virtual and real elements at the algorithm level. As a result, they suffer from imbalanced performance indicators, difficulties in cross-platform usage, and long development cycles. Moreover, existing AR maps use computer-vision-based methods to superimpose prefabricated information as layers on paper maps. This leads to a lack of individual characteristics of each prefab and increases the difficulty in designing dynamic interactive functionality in AR maps.



(a) (b) (c)

Figure 1. AR maps implemented using current mainstream methods. (**a**) AR map implemented using artificial markers. (**b**) AR map implemented using AR system development framework. (**c**) AR map implemented using natural features.

There are many commercial-grade AR SDKs (such as Vuforia [17], EasyAR [18], and ARCore [19]) and AR development software (such as Unity3D [20] and UE4 [21]) available for rapidly building high-performance AR systems. These AR SDKs encapsulate various 3D registration and tracking algorithms for different scenarios and optimize them effectively. Developers only need to predesign AR maps in AR development software. The system will automatically invoke 3D registration and tracking algorithms and adopt cross-platform technology to enable developers to deploy the AR system on multiple devices. Additionally, as these AR development software provide comprehensive data management models and powerful development toolkits, they greatly facilitate the design and development of dynamic interactive functions of AR systems. Many scholars have used such software to develop AR systems applicable to their research scenarios [22–24]. However, mainstream AR development software lack a data conversion interface between existing geographic information data and virtual information data, which hinders the full utilization of existing geographic information data. Additionally, when designing AR maps using such software, the predominant approach is manual prefabrication, which mainly involves manual image recognition. It leads to repetitive and tedious prefabrication work, resulting in errors in the categorization and positioning of prefabs.

The emergence of object detection methods provides a new approach to replace manual image recognition. With the development of object detection methods from traditional computer-vision-based approaches [25–29] to deep learning-based object detection models [30–35], object detection methods have attracted increasing attention from researchers in the field of image recognition due to their high accuracy and fast recognition speed. Song et al. [36] proposed an improved method for map point symbol recognition based on improved generalized Hough transform and non-line AR mapping, which achieved high accuracy in symbol category recognition and positioning. Huang et al. [37] used an improved YOLOv4 object detection model to recognize point symbols in scanned topographic maps, which has higher accuracy and efficiency. Furthermore, in recent years, a new wave of outstanding deep learning models has emerged in the field of object detection. These include end-to-end object detection models such as DINO [38], based on the DETR architecture; DiffusionDet [39], which pioneers the application of diffusion models in object detection frameworks; EfficientDet [40], a scalable detection architecture designed under practical computational resource constraints with higher precision and efficiency; and the latest version of the YOLO series, YOLOv8 [41], among others. While these models have distinct architectural designs, they all demonstrate extremely high levels of accuracy and have achieved prominent rankings within the field. In this paper, we apply the object detection model to map point symbol recognition, which can replace manual map reading, reduce errors, and improve work efficiency when prefabricating AR maps.

In order to reduce the labor-intensive work and inaccuracies in information accuracy associated with manually prefabricating AR maps in existing AR development software, we used Amap [42] as the data source (Amap, also known as Gaode Maps, is a mapping product designed by Gaode Software Co., Ltd. (Beijing, China) to provide high-quality geographic information data and services, and is popular among users for its accurate, real-time data updates and rich features), and taking the map point symbols as an example, designed a method for automatically generating prefabs for AR maps based on the predicted bounding boxes of the object detection model. First, we compared and analyzed various object detection models for recognizing map point symbols. Then, we selected the best-performing object detection model to automatically generate prefabs for AR maps. Finally, we developed an AR map information query prototype system based on the generated prefabs. This study aimed to generate prefabs for point symbols on AR maps automatically, optimize the prefabrication process, and provide higher-quality data support for AR maps.

2. Materials and Methods

In the current AR map prefabrication process, the alignment of the prefabs with the map is mostly conducted manually by hand, which is time-consuming and labor-intensive and is very prone to manual errors. In order to solve the drawbacks of the current methods, we designed a method that can automatically generate prefabs with category and location information. This method can replace the current manual prefabrication process, leading to improved efficiency and reduced errors. This section introduces the proposed method in this paper from four modules: the production of the map point symbol dataset, map point symbol recognition based on an object detection model, the automatic generation method of AR map point symbol prefabs, and the development of an AR map prototype system. Among them, the map point symbol recognition part selected four object detection models that were mainstream and with good performance for comparative analysis. Then, based on the object detection model with the best performance, a series of prefabs with corresponding categories and positional information were automatically generated. Lastly, the AR map prototype system was developed by combining the prefabs with the current mainstream AR system development framework. The detailed schematic is shown in Figure 2.

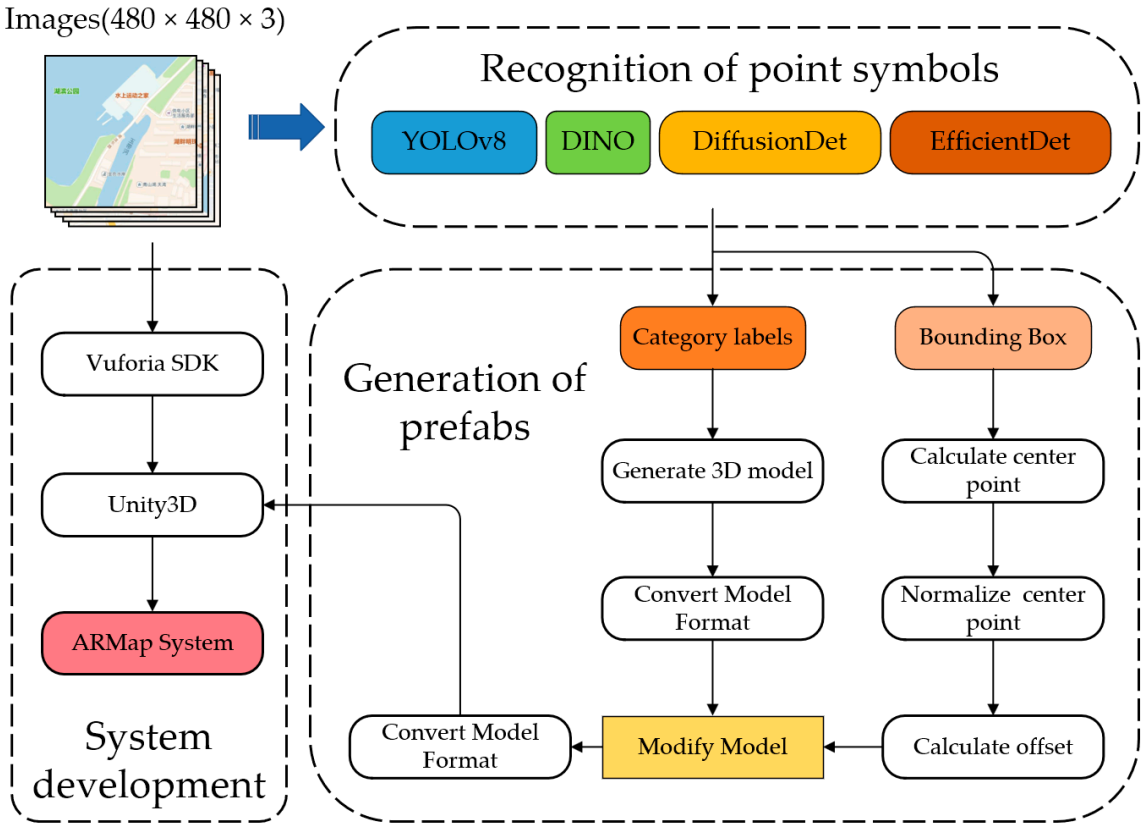


Figure 2. The framework of generating AR map point symbol prefabs based on object detection model. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

2.1. Map Point Symbol Datasets

The data used in this paper were downloaded from Amap and cover the main urban area of Nanchang City, Jiangxi Province, China. We chose five categories of point symbols commonly used in people's daily lives when using maps as the targets of augmented representations: school, hospital, restaurant, shop, and residential area, as exemplified in Table 1.

Table 1. Legend of the five categories of map point symbols.

Symbol					
Description	Shop	Restaurant	School	Residential area	Hospital

First, we divided the downloaded map into 1134 samples of 480 × 480 pixels each, as shown in Figure 3. Then, we manually annotated 1134 samples using LabelImg. A total of 4427 instances of the 5 categories of point symbols were annotated, with the specific distribution shown in Figure 4. Finally, we divided the annotated samples into a training set, validation set, and test set, with the training set accounting for 80% of the total samples, totaling 882 samples. The validation and test sets accounted for 10% of the samples, amounting to 252. The dataset contained map image samples and corresponding txt files, where the txt files included cls_id (numeric encoding of symbol categories), x', y' (the center coordinates of the bounding box, expressed as proportions relative to the width and height of the image), and w', h' (the width and height of the bounding box, expressed as proportions relative to the width and height of the image).

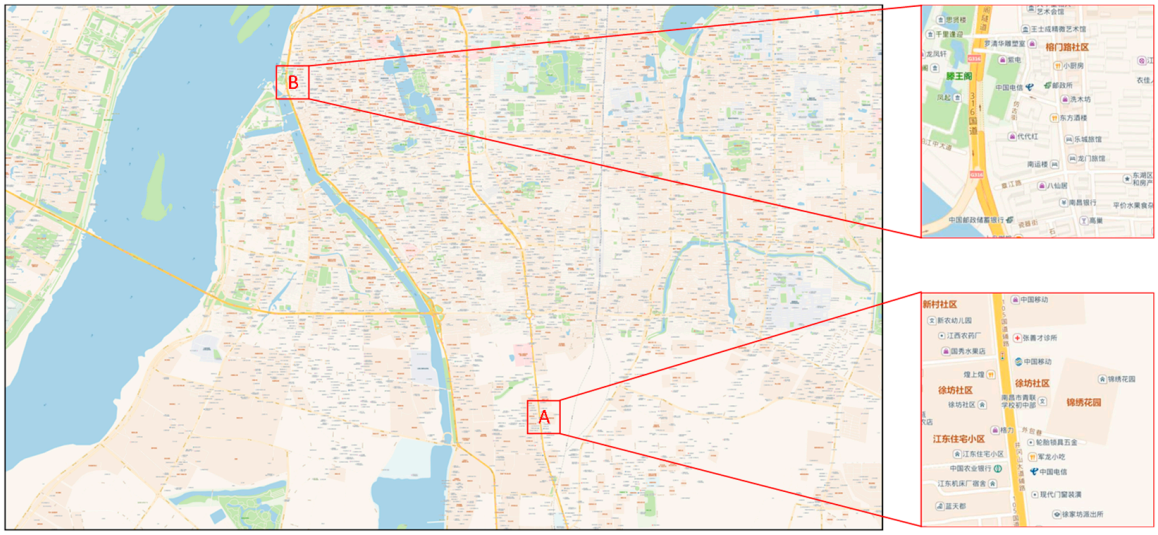


Figure 3. Map of Nanchang City and some map samples. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

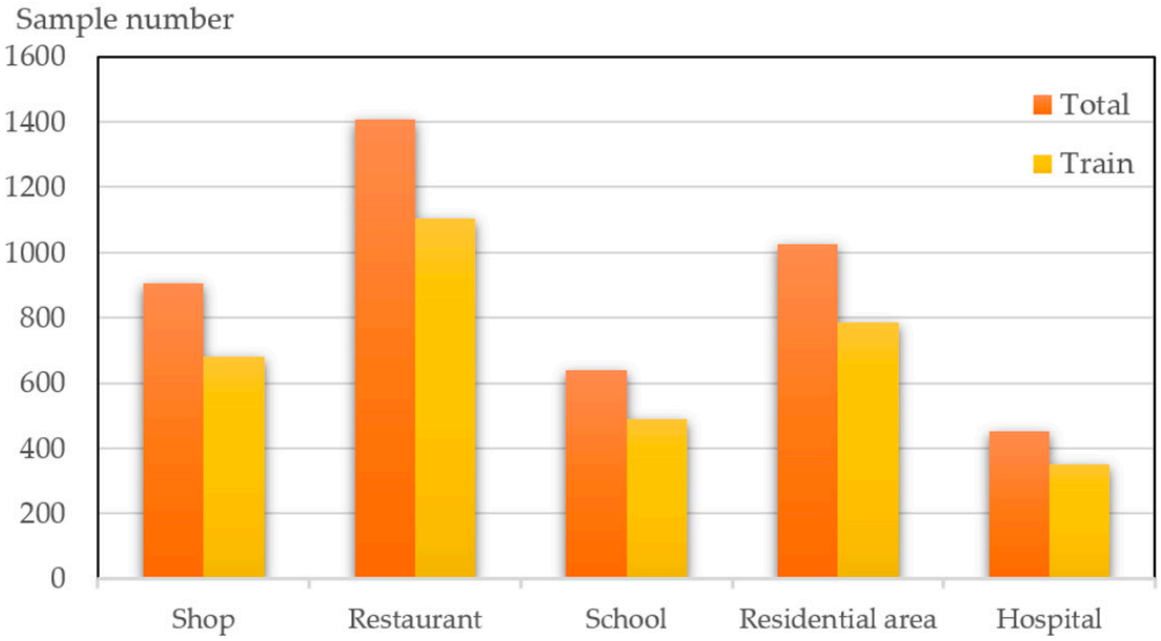


Figure 4. The sample number of five categories of map point symbols in the training set and total set.

2.2. Map Point Symbol Recognition Based on Object Detection Model

Maps usually contain many point symbols consisting of different shapes, texts, and colors, providing rich graphical features and unique semantic information. These point symbols convey a large amount of geographic information to map users. However, the wide variety and dense distribution of point symbols in maps can easily cause omissions when prefabricating AR maps. In addition, as the input data for the method proposed in this paper, the recognition effect of the map point symbols will directly affect the generation of the AR map point symbol prefabs. Therefore, accurately identifying map point symbols is the first research focus of this paper. We intend to utilize an object detection model for the recognition of map point symbols. When the object detection model completes its prediction, it generates bounding boxes with category labels. The coordinates of these bounding boxes along with their label information serve as the specific input data format for the method proposed in this paper.

We selected four mainstream models that have demonstrated excellent performance in the field of object detection. These models have different architectures and are representative in their own right. To determine which model is more suitable for the recognition task in this paper, we conducted a comparative analysis of their performance in map point symbol recognition. The selected models are DINO, DiffusionDet, EfficientDet, and YOLOv8. The YOLOv8 series includes five sub-models: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. Among these, YOLOv8x has the slowest detection speed but the highest mAP (mean average precision) [43]. Therefore, we chose to compare YOLOv8x with the other models. Similarly, within the EfficientDet series, there are eight sub-models labeled D0 to D7. We chose EfficientDet-D0.

YOLOv8 is the latest version of the YOLO (You Only Look Once) series of object detection and image segmentation models developed by Ultralytics, and it is also the version with the best performance in the YOLO series; it consists of the backbone module for extracting features, the neck module that adopts a PAN (path aggregation network) structure, and the head module that generates prediction bounding boxes and output vectors. The specific structure of the YOLOv8 model is shown in Figure 5. YOLOv8 adopts the C2f module, which is richer in gradient streams, backbone, and neck, and adjusts the number of channels differently for different scales of the model. A decoupled head structure is adopted in the head, which mainly includes loss calculation and object detection frame screening. Among them, loss calculation includes positive and negative sample allocation strategy and loss calculation. Task-aligned assigner matching is adopted in the positive and negative sample allocation strategy. BCE loss (binary cross entropy loss) is used in the classification branch of loss calculation, and DFL (distribution focal loss) and CIOU (complete intersection over union) loss functions are used in the regression branch [44].

On the other hand, we evaluated the performance of the object detection model using five evaluation metrics, namely precision (P), recall (R), F1-score ($F1$), mAP_{50}, and $mAP_{50–95}$. This process aims to obtain more comprehensive and objective evaluation results regarding the model's recognition effectiveness. The specific calculation methods for each metric are outlined in Equations (1)–(5).

$$P = \frac{TP}{TP + FP} \times 100\% \tag{1}$$

$$R = \frac{TP}{TP + FN} \times 100\% \tag{2}$$

$$F1 = \frac{2 \times P \times R}{P + R} \times 100\% \tag{3}$$

$$AP = \int_0^1 P(R)dR \times 100\% \tag{4}$$

$$mAP = \frac{\sum_{i=1}^{n} AP_i}{n} \times 100\% \tag{5}$$

where P is precision, R is recall, TP is the number of true positive bounding boxes detected when the confidence score is higher than the predefined threshold, FP is the number of true positive bounding boxes detected when the confidence score is lower than the predefined threshold, FN is the number of true positive bounding boxes that were not detected, $F1$ is the harmonic mean of precision and recall, AP is the average recognition accuracy for a specific category, mAP is the mean average precision, and n is the number of categories.

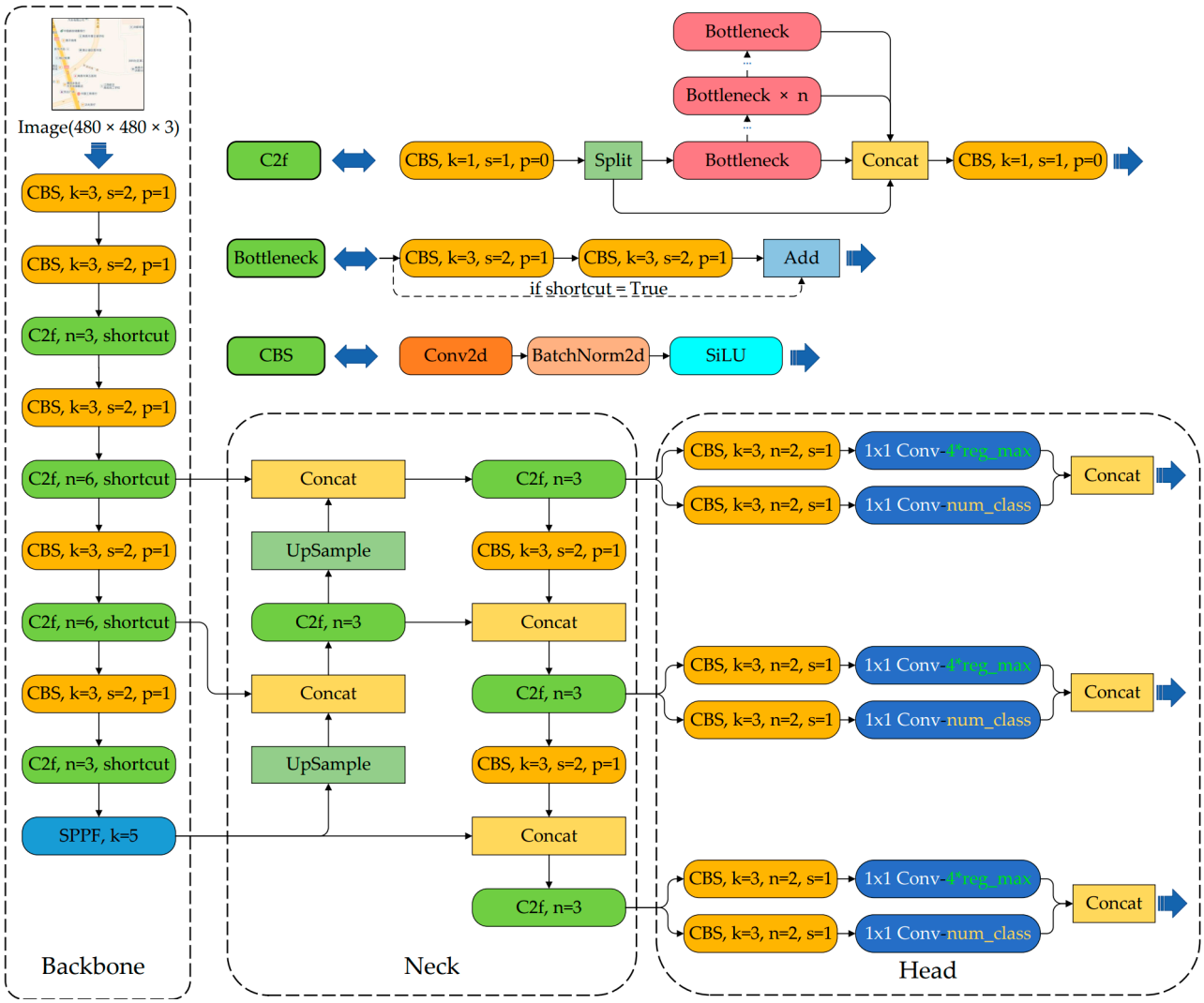


Figure 5. Schematic of the YOLOv8 model structure. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

2.3. An Automated Method for Generating AR Map Point Symbol Prefabs

Due to the lack of proper geographic information data interfaces in existing mainstream AR development software, the reuse of geographic information data is not achievable in the process of prefabricating AR maps. To solve this problem, we proposed an automated method for generating prefabs of AR map point symbols, taking the Unity3D platform as an example. We established a data conversion interface between paper map point symbols and AR map point symbol prefabs. We also transformed the map point symbol information extracted from the object detection model into corresponding 3D models with category and positional information. Then, these 3D models were used as prefabs in AR maps, enabling the reuse of geographic information data and simplifying the prefabrication process of AR maps.

The principle of the automated method for generating prefabs of AR map point symbols was to use the predicted bounding boxes of the object detection model as input data. The category labels of the predicted bounding boxes were used to automatically generate 3D models of the corresponding category. The geometric center of the predicted bounding boxes was then abstracted as the coordinate point of the point symbol in the image pixel coordinate system. Based on the conversion relationship between the image pixel coordinate system and the virtual environment coordinate system, the corresponding point of this coordinate point in the virtual environment coordinate system was obtained.

Finally, the generated 3D models were repositioned using this information, and prefabs with corresponding categories and positional information were output. The detailed schematic is shown in Figure 6.

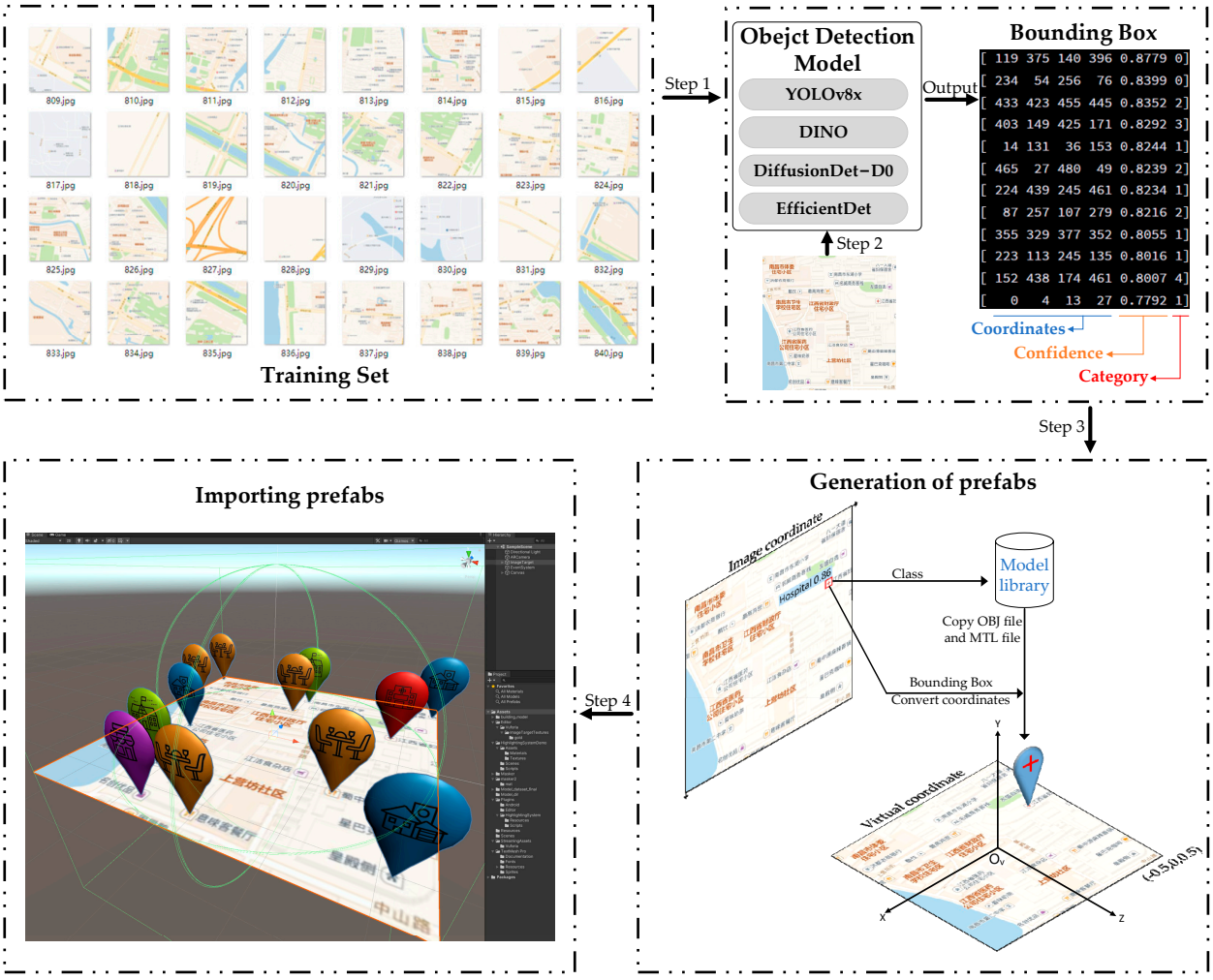


Figure 6. Schematic of the automated method for generating prefabs of AR map point symbols. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

We used the conversion relationship between the image pixel coordinate system and the virtual environment coordinate system to achieve the transfer of information from paper maps to AR maps. Among them, OBJ model files, as a standard 3D model file format, are widely used by various 3D software for 3D model storage and are compatible with most AR development software. They are simple text files that can be used for editing and modification operations without complex data processing. We chose the OBJ model file format as the output method for the prefabs. In Unity3D's virtual coordinate system, the long edge of the image is quantized to a unit of 1, while the short edge is quantized based on the ratio of the short edge to the long edge. If the dimensions of the image are the same, both edges are quantized to a unit of 1. The coordinate origin is set at the center of the image and the initial coordinates of the OBJ models are assumed to be at the origin of the virtual coordinate system. In order to convert the center of the predicted bounding box

into the positioning point of the OBJ model in the virtual coordinate system, we use the formula for calculating the coordinate offset of the OBJ model file, shown in Equation (6).

$$\begin{cases} \Delta x = 0.5 - \frac{x_l + x_r}{2I_l}, & \Delta y = \frac{I_w - y_l - y_r}{2I_l} & I_l > I_w \\ \Delta x = 0.5 - \frac{x_l + x_r}{2I_l}, & \Delta y = 0.5 - \frac{y_l + y_r}{2I_w} & I_l = I_w \\ \Delta x = \frac{I_w - y_l - y_r}{2I_l}, & \Delta y = 0.5 - \frac{y_l + y_r}{2I_w} & I_l < I_w \end{cases} \tag{6}$$

where Δx and Δy are the offset on the x-axis and y-axis, respectively. x_l and y_l are the coordinates of the upper left corner of the prediction bounding box in the image pixel coordinate system. x_r and y_r are the coordinates of the lower right corner of the prediction bounding box in the image pixel coordinate system. I_l and I_w are the length and width of the map image, respectively.

Due to the correspondence between the x–y plane of the image pixel coordinate system and the x–z plane of the virtual environment coordinate system, the coordinate offset obtained from Equation (6) was used to sequentially adjust the coordinate values of the vertex coordinates in the x-axis and z-axis of the OBJ model file. The specific calculation methods are described in Equations (7) and (8). The modified vertex coordinate format is "[V V_{newx} $V_{oldy}V_{newz}$]", where the corrected results replace the original vertices.

$$V_{newx} = V_{oldx} + \Delta x \tag{7}$$

$$V_{newz} = V_{oldz} + \Delta y \tag{8}$$

where V_{newx} and V_{newz} are the adjusted vertex coordinates. V_{oldx} and V_{oldz} are the original vertex coordinates before adjustment. Δx and Δy are the offset amounts.

The specific procedure of the automated method for generating prefabs of AR map point symbols is shown in Table 2.

Table 2. The procedure of the automated method for generating prefabs of AR map point symbols.

Method:	The automated method for generating prefabs of AR map point symbols.
Input:	The predicted bounding of the object detection model during inference
Output:	Prefabs in OBJ file format
1:	Initialize a 2D array data_temp and a constant k
2:	while (k < The number of predicted bounding boxes) do
3:	Read the coordinates and category labels of the bounding box and append them to data_temp
4:	k++
5:	end while
6:	for (i = 0; i < data_temp.length—1; i++) do // Iterate through data_temp
7:	Generate the corresponding OBJ model and MTL material file based on the category label of data_temp[i]
8:	Rename the OBJ file as "category_name + i" and change the file extension to ".txt"
9:	Calculate the offset of the OBJ model based on the coordinate information in data_temp[i], as described by the formula in Equation (6)
10:	Read the vertex coordinates of the model from the OBJ file generated in step 7, and adjust the vertex coordinates based on the offset. The calculation method is described in Equations (7) and (8)
11:	Organize the adjusted vertex coordinates in the "[V V_{newx} $V_{oldy}V_{newz}$]" format and replace the original vertex coordinates with them
12:	Change the file extension of the corrected OBJ model text file to ".obj"
13:	end for

2.4. Development of an AR Map Prototype System

In order to verify the feasibility and practicality of the proposed method, an AR map prototype system was developed. An interactive information query function was

developed within the system. Given the availability of mature commercial-grade AR system development frameworks, Vuforia and Unity3D have been favored by many developers due to their rich functions, high-quality augmented reality integration effects, excellent cross-platform compatibility, scalability, and comprehensive development toolkits. Therefore, this paper adopted the mainstream AR system development framework to develop the AR map prototype system. Unity3D (version: 2019.3.2f1) was selected as the core platform for the development of the system. Additionally, we introduced the necessary AR components in the system with Vuforia SDK (version: 8.5.9). Finally, the system was deployed on the Android mobile platform using the Android NDK (version: r19c).

3. Results

3.1. Recognition of Map Point Symbols

In order to comprehensively analyze the results of the four object detection models selected in this paper, we demonstrate the result of the four object detection models in the map point symbol recognition task in terms of performance metrics, recognition result, and training process. In addition, the training and validation of the four object detection models were conducted using the Windows 11 operating system. The mainstream deep learning framework PyTorch 1.12.1 was used, with the development environment set to Python 3.8. The graphics card used was NVIDIA GeForce RTX 3080 Laptop 32G, and GPU acceleration was achieved using CUDA 11.4 and cuDNN 8.4.1.

During model training, the input sample image size was 480 × 480, the batch size was 16, and a total of 135 iterations were performed. Moreover, the initial learning rate was set to 0.01, the weight decay coefficient was set to 0.0005, the confidence threshold was set to 0.5, and the momentum was set to 0.937. In addition, four object detection models were compared under the same parameter conditions. No pre-training model was used, all four models were trained from scratch.

3.1.1. Visualization of Map Point Symbol Recognition Results

The recognition results of the four object detection models on some samples from the validation set are shown in Figure 7. Figure 7a–d, respectively, show the recognition results of DiffusionDet, DINO, EfficientDet-D0, and YOLOv8x on sample A. Figure 7e–h show the recognition results for sample B.

Figure 7. *Cont.*

Figure 7. Recognition results of DiffusionDet, DINO, EfficientDet-D0, and YOLOv8x on samples A and B. (**a**–**d**) The recognition result of the four object detection models on sample A; (**e**–**h**) the recognition results for sample B; the cut-off point symbol is marked with a red box. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

As shown in Figure 7, four object detection models exhibited high accuracy in recognizing map point symbols. The confidence scores were also generally high. However, both DiffusionDet and YOLOv8x were able to detect a small number of point symbols when they were partially cut off by the edges of the map image tiles, as shown by the red boxes in Figure 7e,h. However, DINO and EfficientDet-D0 failed to recognize such symbols in the same cases. This indicated that YOLOv8x and DiffusionDet had stronger feature learning capabilities and, hence, better recognition performance for map point symbols. However, the confidence scores of DiffusionDet were generally lower than those of YOLOv8x.

3.1.2. Comparison of Performance Metrics of Object Detection Models in Map Point Symbol Recognition

The performance metrics of the four models on the validation set are presented in Table 3. YOLOv8x achieved the highest values in all metrics and is the best-performing model among the four detection models. YOLOv8x outperforms DiffusionDet by about 10.8% in recall, followed slightly by DINO and EfficientDet-D0. Additionally, EfficientDet-D0 outperformed DINO by approximately 0.6% and 1.3% in precision and $mAP_{50–95}$, respectively. However, DINO demonstrated a lead over EfficientDet-D0 by approximately 1.0%, 0.2%, and 0.4% in recall, F1-score, and mAP_{50}, respectively. DiffusionDet lagged behind the other three models on almost all metrics, only marginally outperforming DINO on the precision and $mAP_{50–95}$ metrics by 0.1% and 0.8%, respectively.

Table 3. Performance metrics of the four models for symbol recognition.

MODEL	Precision (%)	Recall (%)	F1-Score (%)	mAP_{50} (%)	$mAP_{50–95}$ (%)
DINO	98.5	97.8	98.1	99.1	68.5
DiffusionDet	98.6	88.3	93.2	98.6	69.3
EfficientDet-D0	99.1	96.8	97.9	98.7	69.8
YOLOv8x	99.6	99.1	99.3	99.3	74.7

As shown in Table 4, DINO demonstrated the best recognition performance in the school and hospital symbol categories, achieving AP values of 100% and 99.9%, respectively. EfficientDet-D0 achieved the best recognition performance in the school, restaurant, and shop symbol categories, with 100% AP achieved in both the school and restaurant symbol categories. YOLOv8x exhibited the best recognition performance in the residential area symbol category, with no less than 98.8% AP across all symbol categories, and had the highest mAP of 99.3%, outperforming the other three models. However, DiffusionDet did not achieve the best performance in any symbol category.

Table 4. AP and mAP values for the four object detection models in each symbol category.

MODEL	School	Residential Area	Restaurant	Hospital	Shop	mAP
DINO	100.0	98.4	98.8	99.9	98.3	99.1
DiffusionDet	99.5	98.1	99.4	99.4	96.4	98.6
EfficientDet-D0	100.0	98.2	100.0	98.2	99.3	99.1
YOLOv8x	99.5	99.2	99.5	99.5	98.8	99.3

The PR curves in Figure 8 show that when the recall is lower than 0.7, the precision of all four models is close to 1. However, when the recall is greater than 0.8, the precision starts to decrease. In general, the closer the curve is to the coordinate (1, 1) position, the better the performance of the model. The area enclosed by the PR curve and the horizontal and vertical coordinates is the mAP. Therefore, as can be seen in Figure 8, the YOLOv8x model has the highest detection accuracy and the best performance.

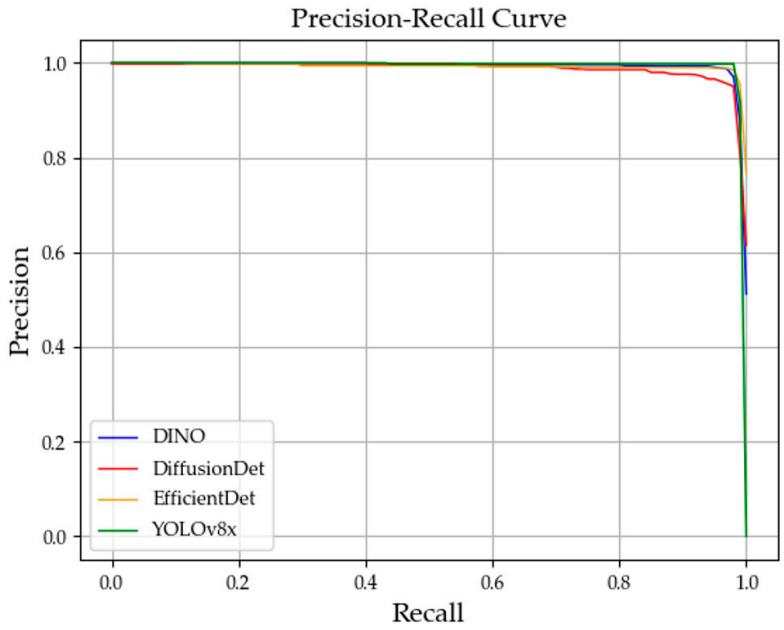
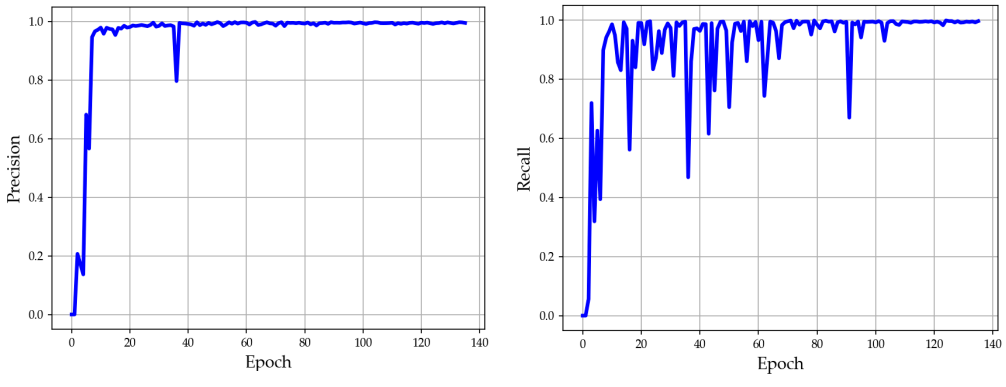


Figure 8. PR curves for four object detection models.

In conclusion, we chose YOLOv8x for map point symbol recognition because it provides reliable data support for the generation of AR map point symbol prefabs. The predicted bounding boxes generated during the object detection model prediction process will serve as input for the automated method for generating prefabs of AR map point symbols.

3.1.3. Training Process of YOLOv8x

The precision, recall, mAP_{50}, and $mAP_{50–95}$ curves during the YOLOv8x training process are shown in Figure 9a–d. They indicate that all the metrics exhibited rapid improvement in the first 20 training epochs. After the 105th training epoch, the precision, recall, and mAP_{50} curves were close to 1, and the $mAP_{50–95}$ curve was close to 0.75. After that, the curve increased slightly. Generally, too few training epochs lead to the underperformance of the model, while too many training epochs lead to overfitting. Therefore, we chose the model output after the 135th training epoch as the object detection model for map point symbols.



(**a**) (**b**)

Figure 9. *Cont.*

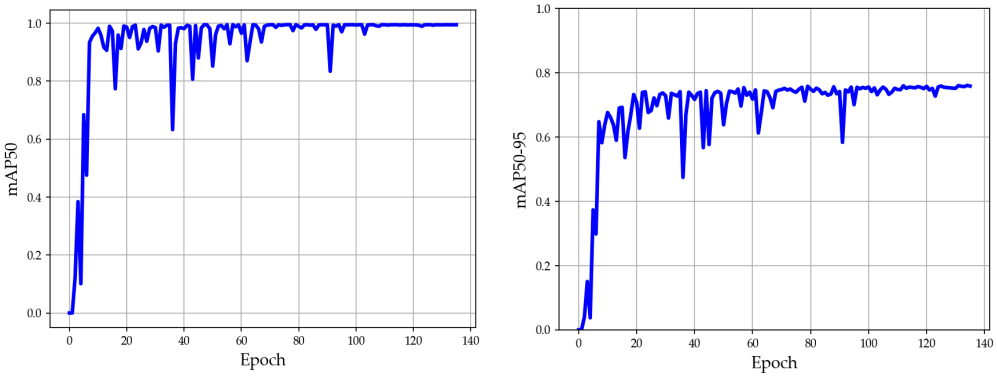


Figure 9. Training process of YOLOv8x. (**a**) Precision; (**b**) recall; (**c**) mAP_{50}; (**d**) $mAP_{50–95}$.

As shown in Figure 10, all three loss curves of YOLOv8x are nearly converged at the end of the training, and there is almost no gap between the 'train' curve and the 'validation' curve, with the latter even showing a slight decline. In addition, we conducted a large number of tests on the validation set and test set, and the test results show that the trained YOLOv8x is a reliable model.

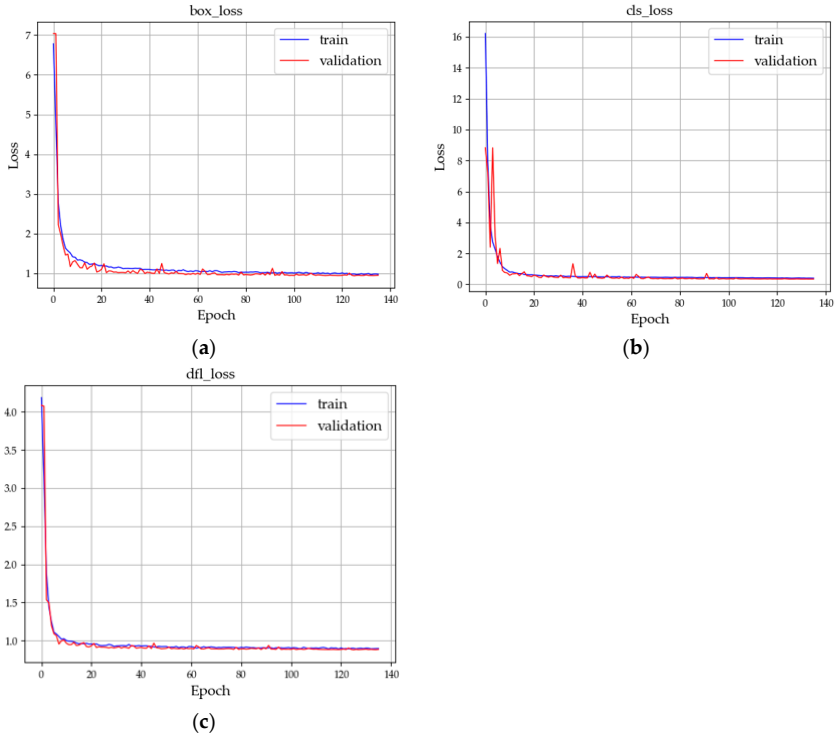


Figure 10. Loss curves of YOLOv8x. (**a**) box_loss; (**b**) cls_loss; (**c**) dfl_loss.

3.2. Automated Generation of Prefabs

3.2.1. Input Data

The input data for the automated method for generating prefabs of AR map point symbols included a series of predicted bounding boxes generated by the object detection model during the prediction phase. As shown in Table 5, these data included the coordinates of the predicted bounding boxes in the image pixel coordinate system (the coordinates of the upper left and lower right corners) and the numerical labels corresponding to the symbol category.

Table 5. Examples of input data for the automated method for generating prefabs of AR map point symbols.

Index	x_l	y_l	x_r	y_r	Confidence	Category
1	353	328	378	353	0.84271	1
2	13	130	37	154	0.82989	4
3	221	111	245	136	0.82106	2
4	222	437	246	463	0.81471	1
5	234	53	257	77	0.55326	0
...	...	...	...	...	...	...
12	0	4	13	27	0.77921	1

3.2.2. Visualization of the Prefabs Generated through the Automatic Method for Augmented Map Point Symbol Prefabs

The predicted bounding boxes of the object detection model were used as input data for the automated method for generating prefabs of AR map point symbols. After processing, the method output a series of OBJ models corresponding to the categories of map point symbols, and their respective positional information. As shown in Figure 11, these models were then loaded into the virtual environment of Unity3D. Furthermore, all the point symbols for augmented representation were generated as prefabs, and the five categories of prefabs generated were accurately aligned with the base map in terms of category and position.

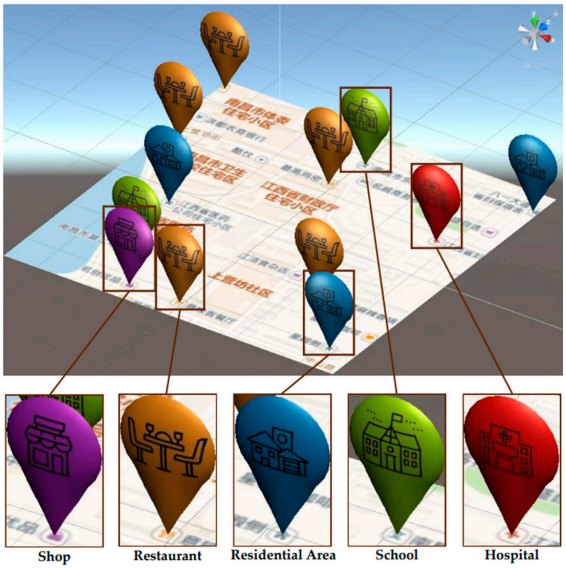


Figure 11. The visualization result of the automated method for generating prefabs of AR map point symbols. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

3.3. System Development and Method Validation

We integrated the generated prefabs into the AR system development framework, designed a database to store additional attribute information of the prefabs, and finally developed an AR map prototype system. In addition, we designed interactive functions for information querying in the system, which allows users to click on the prefab to query the additional attribute information, in order to validate the functionality and effectiveness of the automated method for generating prefabs of AR map point symbols, and to demonstrate that the automatically generated prefabs are visual markers with interactive potential.

Firstly, we uploaded the map image to the Vuforia official website. We then extracted its image features into a data package using the website tool and loaded it into Unity3D. Then, we introduced the Vuforia AR component to import the map image into the virtual

environment coordinate system and loaded the generated prefabs into the virtual environment. Finally, the Android NDK was used to deploy the AR map prototype system to the validation device. The system environment of the validation device is shown in Table 6.

Table 6. System environment of the validation device.

System Environment	Reference Value
Operating System	HarmonyOS 2.0
Processor	Huawei Kirin 820
RAM	6.0 GB
Screen Size	2000 × 1200
Camera Pixel	3264 × 2448

The proposed method in this paper was validated using the AR map prototype system. The operation effect graph of the AR map prototype system is shown in Figure 12. This shows that the real-time performance and stability of this system met the application requirements of the AR map on the validation device used in this paper. The system validation results indicated that the automated method for generating prefabs of AR map point symbols could be used in the prefabrication process of the AR map. Combined with the object detection model, this method could automatically generate AR map prefabs with positional information, replacing the previous manual prefabrication method. It ensured high information accuracy while simplifying the prefabrication process of the AR map, demonstrating practicality and feasibility. In addition, users could select prefabs by touching the screen, and when a prefab was selected, the information box on the left side of the system page displayed additional information about that prefab. This proved that the automated method for generating prefabs of AR map point symbols could provide solid data support for the development of dynamic interactive functions in the AR map.

(**a**)

(**b**)

Figure 12. Visualization of the running effects of the AR prototype system developed in this paper. (**a**) A screenshot of the AR map prototype system in operation. (**b**) A photo of the AR map prototype system running on the validation device. (This paper uses Amap as an example, in which the Chinese characters do not affect the experimental results).

4. Discussions

Investigating methods for automatically generating AR map prefabs has a positive impact on optimizing the prefabrication process of AR maps. This paper explored the feasibility of integrating object detection models for automatically generating AR map prefabs. We made valuable contributions to optimizing the AR map prefabrication process by ensuring extremely high information accuracy while simplifying the process. Additionally, our approach provided better data support for the dynamic interactive functions of AR maps, demonstrating the feasibility of our design. The main contributions of this paper can be summarized as follows: (1) We constructed a dataset of map point symbols and compared the recognition performance of four object detection models on map point symbols. (2) We proposed a method for automatically generating AR map point symbol prefabs and established an interface for converting map point symbols to AR map point symbol prefabs. (3) We integrated the AR system development framework to design an AR map prototype system.

Even though the precision and recall of YOLOv8x on the validation set are as high as 99.6% and 99.1%, respectively, for large map data, minor imperfections may arise, which in turn affect the accurate generation of prefabs. In order to correct these minor errors, some manual intervention may still be required. Regardless, the method proposed in this paper greatly improves the prefabrication efficiency and information accuracy of AR maps compared to previous manual prefabrication methods.

According to the results of the automated generation of prefabs, our proposed method led to overlap and coverage between prefabs in areas where map point symbols were densely distributed. The reason might be that paper maps can only synthesize map symbols in two-dimensional space, while AR maps are dimensional representations based on paper maps, which lack automatic synthesis in three-dimensional space like electronic maps. Therefore, it led to the information being covered in three-dimensional space, adversely affecting interactive functionality use. In future research, we will develop a set of prefab symbols at different scales inspired by the design principle of cartographic synthesis, aiming to introduce cartographic synthesis rules into AR maps.

Since the prefabs generated in our method were individual data management objects with positional information, we not only greatly simplified the prefabrication process of AR maps but also provided solid data support for the dynamic interaction function and spatial analysis function of AR maps. In the AR map prototype system, we designed an interactive information query function so that users could click on prefabs to view more information about the corresponding symbols, effectively expanding the information capacity of paper maps. In future research, we can add more attribute information to the generated prefabs and combine them with interactive functionalities to design more advanced spatial analysis functions. The spatial analysis results will be loaded into AR maps in the form of prefabs, further enhancing the practicality of AR maps.

However, the method proposed in this paper considered only point symbols, lacking considerations for the augmented representation of other map elements. However, for a complete AR map system, it is not enough to focus on the augmented representation of map point symbols. There are also many line symbols, polygon symbols, annotation information, and other map elements that have the potential for augmented representation that can enrich the AR maps. From these perspectives, relying only on point symbol prefabs in AR maps limits the expansion of map information and weakens their practicality. Despite these limitations, this paper mainly focuses on exploring the feasibility of transforming paper map symbol elements into AR map prefabs by integrating object detection models. In the future, we plan to integrate additional map elements into the automatic generation method of AR map prefabs to further improve the data conversion interface from paper maps to AR maps. Moreover, we intend to include more map symbols of various styles in the training set of the object detection model. This aims to enhance the semantic understanding of map symbols and improve the object detection model's capability to process diverse map data.

5. Conclusions

Addressing the problem that existing AR development software mainly rely on manual prefabrication when creating AR maps, this paper mainly focuses on the automatic generation of point symbol prefabs in AR maps. First, the effectiveness of four object detection models—DINO, DiffusinDet, EfficientDet-D0, and YOLOv8x—were compared and analyzed. The model with the best performance was selected for recognizing map point symbols. Then, based on the predicted bounding boxes of the object detection model, we proposed a method to generate AR map point symbol prefabs automatically. Last, combining the generated prefabs, an AR map prototype system was developed for the Android platform using the Unity3D+Vuforia+Android development framework. The results were as follows:

Firstly, the YOLOv8x model achieved 99.6% precision and 99.1% recall on the validation set, completing the recognition of a sample in an average of 29.4 ms. It outperformed the other three models and was significantly superior to manual image recognition.

Secondly, the prefabs generated by the proposed method in this paper correspond to the categories of map point symbols and include positional information. The prefabs can accurately align with the map point symbols when imported into the virtual environment. Furthermore, the prefabs generated by the method were individual data management objects, providing solid data support for the design of dynamic interactive and analytical functions in AR maps.

Finally, the virtual reality fusion effect of the AR map prototype system developed in this paper is stable and accurate. Moreover, the method proposed in this paper can be applied to actual AR map prefabrication processes, significantly reducing the workload while maintaining high accuracy. In addition, the information query function was realized by clicking on the prefabs through the touchscreen interaction, expanding the information capacity beyond that of traditional paper maps.

Author Contributions: Nixiao Zou completed most of the experiments and analyzed the results. Qing Xu provided crucial suggestions for modifications to this research work and inspected the quality of the experiments conducted in this paper. Yuqing Wu participated in the construction of the map point symbol dataset. Xinming Zhu provided the necessary experimental equipment mentioned in this paper and partial financial support. Youneng Su contributed to the development of the AR map prototype system. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Fund Project of ZhongYuan Scholar of Henan Province of China under Grant [number 202101510001].

Data Availability Statement: The data presented in this study are available from the author upon reasonable request.

Acknowledgments: The authors are grateful to the editors and the anonymous referees for their valuable comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. An, Z.; Zhuang, J.; Qi, Q. Visualization and interaction of augmented paper maps based on augmented reality. *Trop. Geogr.* **2012**, *32*, 476–480.
2. Zhang, J.; Xia, X.; Liu, R.; Li, N. Enhancing human indoor cognitive map development and wayfinding performance with immersive augmented reality-based navigation systems. *Adv. Eng. Inform.* **2021**, *50*, 101432. [CrossRef]
3. Lobo, M.-J.; Christophe, S. Opportunities and challenges for augmented reality situated geographical visualization. *Copernic. GmbH* **2020**, *V-4-2020*, 163–170. [CrossRef]
4. Sun, M.; Chen, X.; Zhang, F.; Zheng, H. Augment reality geographical information system. *Acta Centiarum Nat. Univ. Pekinesis* **2004**, *40*, 906–913.
5. Waldman, I.J. Augmented Reality Maps. U.S. Patent 20110199479, 18 August 2011.
6. Pang, J.; Chen, G.; Song, G.; Zhang, X. Research and application of augmented reality map. *J. Geomat.* **2021**, *46*, 5.
7. Chatain, J.; Demangeat, M.; Brock, A.M.; Laval, D.; Hachet, M. Exploring input modalities for interacting with augmented paper maps. In Proceedings of the 27th Conference l'Interaction Homme-Machine, Toulouse, France, 27–30 October 2015.

8. Bobrich, J.; Otto, S. Augmented maps. *Geospat. Theory Process. Appl.* **2002**, *34*, 502–505.
9. Wang, X.U.; Li, A. Research on paper map augmented reality registration technology. *J. Geomat. Sci. Technol.* **2016**, *33*, 185–190.
10. Wang, Z. An AR Map Virtual–Real Fusion Method Based on Element Recognition. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 126. [CrossRef]
11. Huang, K.; Wang, C.; Wang, S.; Liu, R.; Chen, G.; Li, X. An Efficient, Platform-Independent Map Rendering Framework for Mobile Augmented Reality. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 593. [CrossRef]
12. Luo, T.; Liu, Z.; Pan, Z.; Zhang, M. A virtual-real occlusion method based on gpu acceleration for mr. In Proceedings of the 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 23–27 March 2019; pp. 1068–1069.
13. Han, X.; Liu, H.; Fan, Y. Research on testing and evaluation of usv autonomous navigation algorithms based on virtual reality fusion. In Proceedings of the 2022 41st Chinese Control Conference (CCC), Hefei, China, 25–27 July 2022; pp. 4329–4336.
14. Wu, Y.; Liu, C. A method of aerial multi-modal image registration for a low-visibility approach based on virtual reality fusion. *Appl. Sci.* **2023**, *13*, 3396. [CrossRef]
15. Morrison, A.; Oulasvirta, A.; Peltonen, P.; Lemmel, S.; Juustila, A. Like bees around the hive: A comparative study of a mobile augmented reality map. In Proceedings of the 27th International Conference on Human Factors in Computing Systems, CHI 2009, Boston, MA, USA, 4–9 April 2009.
16. You, S.; Neumann, U. Orientation tracking for outdoor augmented reality registration. *IEEE Comput. Graph. Appl.* **1999**, *19*, 36–42. [CrossRef]
17. Alyousify, A.L.; Mstafa, R.J. AR-assisted children book for smart teaching and learning of turkish alphabets. *Virtual Real. Intell. Hardw.* **2022**, *4*, 263–277. [CrossRef]
18. Liu, Z.; Tang, B.; Cao, S. Design and implementation of the converging media system based on webar. In Proceedings of the 5th International Conference on Information Management (ICIM), Batu, Indonesia, 24–27 March 2019.
19. Feigl, T.; Porada, A.; Steiner, S.; Loeffler, C.; Mutschler, C.; Philippsen, M. Localization limitations of arcore, arkit, and hololens in dynamic large-scale industry environments. In Proceedings of the GRAPP 2020—15th International Conference on Computer Graphics Theory and Applications, Valletta, Malta, 27–29 February 2020.
20. Zheng, L.-G.; Zhu, M.; Yu, H.N. A virtual environment making method for cave system. In Proceedings of the 2020 International Conference on Computing and Data Science (CDS), Stanford, CA, USA, 1–2 August 2020; pp. 377–380.
21. Yuan, J.; Fan, A.; Xing, R. Design and Implementation of Educational VR Games Based on UE4. In Proceedings of the DEStech Transactions on Computer Science and Engineering, Chengdu, China, 29–30 June 2020; pp. 249–254. [CrossRef]
22. Liu, S.; Cao, Y.; Gao, L.; Xu, J.; Zeng, W. Realization of mobile augmented reality system based on image recognition. *J. Inf. Hiding Priv. Prot.* **2021**, *3*, 55–59. [CrossRef]
23. Septiawan, I.D.; Taurusta, C. Application of augmented reality for gypsum marketing using vuforia, sketchup and unity 3D. *Procedia Eng. Life Sci.* **2021**, *1*, 1–5.
24. Liu, X.; Sohn, Y.H.; Park, D.W. Application development with augmented reality technique using unity 3D and vuforia. *Int. J. Appl. Eng. Res.* **2018**, *13*, 15068–15071.
25. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G.R. ORB: An efficient alternative to sift or surf. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
26. Dalal, N. Histograms of oriented gradients for human detection. In Proceedings of the Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–26 June 2005.
27. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
28. Schuldt, C.; Laptev, I.; Caputo, B. Recognizing human actions: A local SVM approach. In Proceedings of the International Conference on Pattern Recognition, Cambridge, UK, 23–26 August 2004.
29. Rätsch, G.; Onoda, T.; Müller, K.-R. Soft margins for adaboost. *Mach. Learn.* **2001**, *42*, 287–320. [CrossRef]
30. Girshick, R. Fast r-cnn. In Proceedings of the International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015.
31. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-stime object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
32. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.E.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Santiago, Chile, 7–13 December 2015.
33. Girshick, R.B.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
34. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
35. Redmon, J.; Divvala, S.K.; Girshick, R.B.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
36. Song, J.; Zhang, Z.; Qi, Y.; Miao, Q. Point symbol recognition algorithm based on improved generalized hough transform and nonlinear mapping. In Proceedings of the 2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), Jinan, China, 14–17 December 2018; pp. 134–139.
37. Huang, W.; Sun, Q.; Yu, A.; Guo, W.; Xu, Q.; Wen, B.; Xu, L. Leveraging Deep Convolutional Neural Network for Point Symbol Recognition in Scanned Topographic Maps. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 128. [CrossRef]

38. Zhang, H.; Li, F.; Liu, S.; Zhang, L.; Su, H.; Zhu, J.-J.; Ni, L.M.-S.; Shum, H.-Y. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv* **2022**, arXiv:2203.03605.
39. Chen, S.; Sun, P.; Song, Y.; Luo, P. Diffusiondet: Diffusion model for object detection. *arXiv* **2022**, arXiv:211.09788.
40. Tan, M.; Pang, R.; Le, Q.V. Efficientdet: Scalable and efficient object detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Washington, DC, USA, 14–19 June 2020; pp. 10778–10787.
41. Sledevič, T.; Plonis, D. Toward bee behavioral pattern recognition on hive entrance using yolov8. In Proceedings of the 2023 IEEE 10th Jubilee Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Vilnius, Lithuania, 27–29 April 2023; pp. 1–4.
42. Huang, Y.H.; Liu, X.P.; Liu, Y.P.; Zhang, H. Spatial and temporal accessibility analysis of urban parks based on amap api by means of multiple transportation: A case study of haizhu district in guangzhou. *Geogr. Geo-Inf. Sci.* **2018**, *34*, 50–57.
43. Zhang, H.; Zhou, X.; Li, H.; Zhu, G.; Li, H. Machine Recognition of Map Point Symbols Based on YOLOv3 and Automatic Configuration Associated with POI. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 540. [CrossRef]
44. Yuan, H.; Tao, L. Detection and idectification of fish in electronic monitoring data of commercial fishing vessels based on improved yolov8. *J. Dalian Ocean. Univ.* **2023**, *38*, 533–542.