Genetic Programming with Aggregate Channel Features for Flower Localization Using Limited Training Data

Qinyu Wang[1], Ying Bi[2](✉), Bing Xue[1], and Mengjie Zhang[1]

[1] Victoria University of Wellington, Wellington 6140, New Zealand
{qinyu.wang,bing.xue,mengjie.zhang}@ecs.vuw.ac.nz
[2] Zhengzhou University, Zhengzhou 450001, China
yingbi@zzu.edu.cn

Abstract. Flower localization is a crucial image pre-processing step for subsequent classification/recognition that confronts challenges with diverse flower species, varying imaging conditions, and limited data. Existing flower localization methods face limitations, including reliance on color information, low model interpretability, and a large demand for training data. This paper proposes a new genetic programming (GP) approach called ACFGP with a novel representation to automated flower localization with limited training data. The novel GP representation enables ACFGP to evolve effective programs for generating aggregate channel features and achieving flower localization in diverse scenarios. Comparative evaluations against the baseline benchmark algorithm and YOLOv8 demonstrate ACFGP's superior performance. Further analysis highlights the effectiveness of the aggregate channel features generated by ACFGP programs, demonstrating the superiority of ACFGP in addressing challenging flower localization tasks.

Keywords: Genetic programming · Aggregate channel features · Flower localization

1 Introduction

Flower localization is a computer vision task focused on precisely determining the location of flowers within images. As a common image pre-processing operation, flower localization contributes to the further identification and classification of flower species, supporting botanical research and biodiversity monitoring [12]. The main challenges of flower localization stem from the diversity of flower classes, variations within specific flowers, various imaging conditions, and limited data on some rare flower species.

To address the challenges above, manual segmentation methods, such as GrabCut [17], are typically used to localize the flower and remove the background as image pre-processing in the early stage, which requires labor-intensive efforts. For automatic flower localization, traditional methods typically utilize flower color and shape information to design threshold-based methods [5,11,18]. However, these methods require domain knowledge about flowers for manual selection of feature descriptors and algorithm design. Furthermore, the performance of common threshold-based techniques based on manually crafted image processing programs might deteriorate when confronted with complex background scenes and diverse imaging conditions in flower images. The aggregate channel features (ACF) detector, which extends channels beyond typical color representations to non-color features, has proven effective in enhancing robustness for localizing objects. It has been successfully applied in various object detection tasks, such as localizing pedestrians [3], faces [22], and urine sediments [19]. However, its application in flower localization remains unexplored.

Convolutional neural network (CNN) methods such as Faster R-CNN [16] and YOLO [6] have been widely applied in object detection, including localizing and classifying flowers [15,20]. However, these methods need pre-trained models and still demand a relatively large amount of training data for fine-tuning due to high model complexity. Their large network structures require Graphics processing units (GPUs) for training, resulting in high computational costs. Additionally, the black-box nature of these models reduces the interpretability in explaining object localization processes.

Genetic Programming (GP) is an evolutionary algorithm inspired by natural selection, designed to automatically evolve programs/solutions through the iterative application of evaluation, selection, and genetic operations including mutation and crossover [8]. The tree-based variable-length representation in GP enables the generation of programs with flexible structures and good interpretability. GP has been successfully applied in image classification with limited data [1].

However, existing GP-based methods that involve region detection [2,21] lack flexibility as the positions of the detected regions are randomly selected and remain fixed across all images. Therefore, the region detection mechanism employed in these methods lacks the capability to effectively capture flowers with varying positions within an image.

1.1 Goals

The goal of this paper is to develop a GP approach, named aggregate channel feature based GP (ACFGP), that can automatically generate effective aggregate channel features to localize the flower within an image with limited training data. This paper focuses on the single-object localization task, where the evolved program is expected to produce a bounding box that accurately locates the flower in each image. The specific objectives of this work are summarised as follows:

- Develop a new function set and a new terminal set that include image processing operators to extract and aggregate channel features to enhance the flower, and dynamically compute bounding boxes based on the specific characteristics of each image;
- Design a new GP representation to evolve effective solutions/programs for various flower localization tasks with limited training data;

– Analyze and compare the performance of ACFGP with the baseline benchmark and the YOLOv8 method; and
– Further analyze the trees evolved by ACFGP along with the visualized results to investigate the effectiveness of aggregate channel features.

2 Backgrounds and Related Work

2.1 Existing Methods for Flower Localization

In the early days, manual segmentation algorithms such as GrabCut [17] are used to segment flowers and remove background [14]. GrabCut is a semi-automatic image segmentation algorithm developed based on the color distribution and graph-cut approach, which requires manually framing the bounding box of the flower as a rough segmentation. An automated flower segmentation algorithm [11] is proposed as a pre-processing for automated flower classification [12]. This method consists of two models, one color model for foreground/background segmentation and a generic shape model for petal structure. However, the performance of color-based segmentation might be affected by diverse illumination situations. Threshold-based algorithms are commonly used to localize flowers as pre-processing in traditional flower classification methods [5,18]. A flower is segmented by the threshold calculated based on the intensity values transformed from color information, whose performance might be affected under complex background and lighting conditions.

Flower localization is commonly regarded as part of the flower detection task within CNN methods, often coupled with classification. In [15], an improved Region Proposal Network (RPN) is utilized to generate region proposals for flowers, and a modified Faster R-CNN model is employed for bounding box regression. In [20], flower localization and classification are simultaneously addressed using the YOLOv4 model [6]. Despite the model's simplification via a channel pruning algorithm, the pruned version still contains over two million parameters, resulting in low interpretability. In addition, both methods in [15,20] require pretrained models, and thousands of flower images are required for fine-tuning.

2.2 Aggregate Channel Features (ACF) for Object Localization

A channel typically refers to a component of an image that represents specific color information, such as red, green, and blue. In this paper, we borrow the concept from [3], where channels are defined as a feature map of the original image, whose pixels are computed from corresponding patches of original pixels. Based on this definition, additional non-color channels are introduced that include a series of feature maps generated through image filters and descriptors. The additional channels enable comprehensive image analysis beyond color information, thus enhancing the capability to comprehend and interpret complex visual data.

ACF methods have been applied to diverse object localization methods. In [3], integral/aggregate channel features are introduced via linear and non-linear transformations. The experimental evaluation demonstrates that ACF enables accurate spatial localization during detection if designed properly. In [22], channel features including color channels, (i.e., Gray-scale, RGB, HSV, and LUV), gradient magnitude, and gradient histograms are used for face detection. Similarly, in [19], ten channel features, i.e., three color channels in LUV space, normalized gradient magnitude, and histogram of oriented gradients (six orientations) are used to localize urine sediments within images. However, these methods rely on a sliding window for localization, which increases the computational cost and limits the ability to localize objects with varying sizes.

In summary, traditional automated flower localization methods primarily rely on thresholding based on color information, making them less effective under complex backgrounds and diverse imaging conditions. CNN-based methods often need fine-tuning with pre-trained models for flower localization, still demanding thousands of images for fine-tuning, and lacking interpretability due to high model complexity. Existing GP-based methods have limitations in flower localization due to randomly selected regions that remain static across all images. Existing ACF methods that rely on sliding windows for localization restrict the adaptability to objects of varying sizes. Therefore, this paper aims to develop a GP approach with ACF to flower localization with limited training data.

3 Proposed Approach

3.1 The New GP Representation

The proposed ACFGP approach has a new GP representation that is based on strongly typed GP (STGP) [10]. In ACFGP, the input for the GP program/tree is an RGB image, and the output is one bounding box identifying the salient flower within the image. ACFGP comprises five layers from input to output, each with a distinct role in the process:

- Input layer: Represents the input RGB image;
- Channel extension layer: Involves color channel selection and channel feature extraction. This layer begins by choosing one from seven color channels (including RGB, LUV, and grayscale) and then applies an image processing filter to the chosen channel, resulting in an extended channel feature.
- Channel aggregation layer: Aggregate the extended channel features from different tree branches to create a saliency mask, highlighting the regions of interest.
- Object localization layer: Detects the salient flower and identifies its bounding box based on the saliency mask.
- Output layer: Represents the bounding box of the flower.

Figure 1 shows how the proposed ACFGP approach constructs a tree using various functions and terminals.

3.2 Terminal Set

Terminals serve as the leaf nodes in a GP tree. There are two types of terminals in ACFGP, i.e., *Img* and *Idx*, as detailed in Table 1. *Img* denotes the RGB image containing salient flowers. It's a three-dimensional array with red, green, and blue pixel values. *Idx* denotes the index used to select a color channel from the RGB image. *Idx* is an integer ranging from 0 to 6, where each index corresponds to one of the seven color channels, i.e., blue, red, green, L, U, V, and grayscale channels.

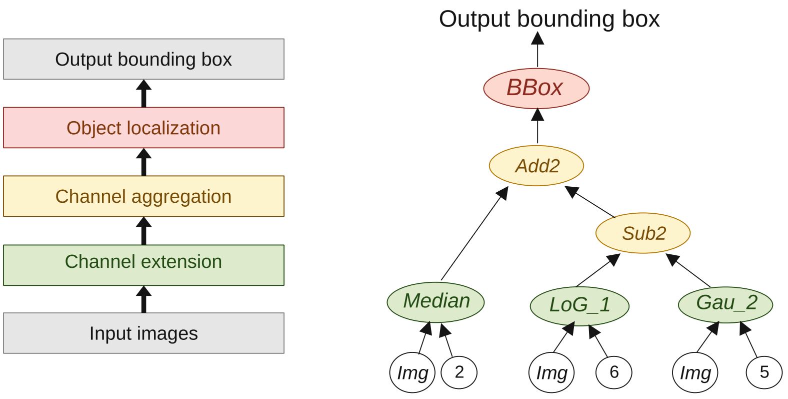

Fig. 1. The program structure and an example program/tree of ACFGP.

Table 1. Terminal Set

Terminal	Type	Value range	Description
Img	array	[0,255]	The 3-channel RGB image
Idx	integer	[0,6]	The index to generate a color channel from an RGB image. Including RGB, LUV, and gray-scale channels

3.3 Function Set

The function set of ACFGP is presented in Table 2 and consists of three categories: channel extension functions, channel aggregation functions, and object localization functions.

Channel extension functions are the image processing filters and descriptors, i.e., *Gau_1*, *Gau_2*, *Median*, *Min_max*, *LoG_1*, *LoG_2*, *Sobel*, *HOG*, and *Saliency*. *Gau_1*, *Gau_2*, and *Median* are used for image smoothing and noise reduction. *Gau_1* and *Gau_2* execute Gaussian filtering with different σ, addressing small/larger scale smoothing. *Min_max*, *LoG_1*,*LoG_2*, and *Sobel* functions are different types of edge detectors. *LoG_1* and *LoG_2* focus on small/larger scale of edge enhancement. *HOG* is a feature descriptor that extracts the distribution and orientation of gradients in an image, capturing important structural information from images, and allowing for robust and reliable pattern recognition. The *Saliency* function is from OpenCV's Saliency module, which detects visually prominent areas in an image based on color, texture, and intensity.

Channel aggregation functions are used to aggregate channel features generated by the channel extension functions into a saliency mask, which is used to localize the salient flower. Since the channel features extracted by the channel extension functions can be considered as a feature map, where pixel values represent feature intensities at specific locations, channel aggregation functions perform addition or subtraction operations on these intensities to enhance features in the region of interest, generating a saliency mask. These functions include *Sub2*, *Add2*, *Add3*, and *Add4*. *Sub2* calculates the absolute difference between two channels, whereas *Add2*, *Add3*, and *Add4* functions perform addition operations on two, three, and four channels, respectively. *Sub2* and *Add2* functions have the flexibility

Table 2. Function Set

Function	Input	Output	Description
Channel extension functions			
Gau_1	*Img, Idx*	*Channel*	Perform Gaussian filtering with stand deviation σ 1 on the selected color channel
Gau_2	*Img, Idx*	*Channel*	Perform Gaussian filtering with stand deviation σ 2 on the selected color channel
Median	*Img, Idx*	*Channel*	Perform 5×5 median filtering on the selected color channel
Min_max	*Img, Idx*	*Channel*	Perform 3×3 min-max filtering on the selected color channel
LoG_1	*Img, Idx*	*Channel*	Perform Laplacian of Gaussian filtering with stand deviation σ 1 on the selected color channel
LoG_2	*Img, Idx*	*Channel*	Perform Laplacian of Gaussian filtering with stand deviation σ 2 on the selected color channel
Sobel	*Img, Idx*	*Channel*	Perform 3×3 Sobel filtering on the selected color channel
HOG	*Img, Idx*	*Channel*	Extract HOG feature vectors on the selected color channel and transfer them into an image-based representation
Saliency	*Img*	*Channel*	Perform static saliency detection and generate a Saliency map
Channel aggregation functions			
Sub2	*2 Channels*	*Channel/Mask*	Perform the absolute difference operation between 2 channels
Add2	*2 Channels*	*Channel/Mask*	Perform the addition operation on 2 channels
Add3	*3 Channels*	*Mask*	Perform the addition operation on 3 channels
Add4	*4 Channels*	*Mask*	Perform the addition operation on 4 channels
Object localization functions			
GBBox	*Mask*	*BBox*	Calculate the bounding box based on the mask with a 5×5 Gaussian filtering
oBBox	*Mask*	*BBox*	Calculate the bounding box based on the mask with an opening operation
cBBox	*Mask*	*BBox*	Calculate the bounding box based on the mask with a closing operation

to produce either an intermediate channel feature for further aggregation or a saliency mask for object localization.

Object localization functions detect the salient flower in an image using the saliency mask and determine the flower's bounding box. The three functions, *GBBox*, *oBBox*, and *cBBox*, employ OpenCV operations including threshold, morphologyEx, findContours, and boundingRect, to execute the object localization process, as shown in Fig. 2.

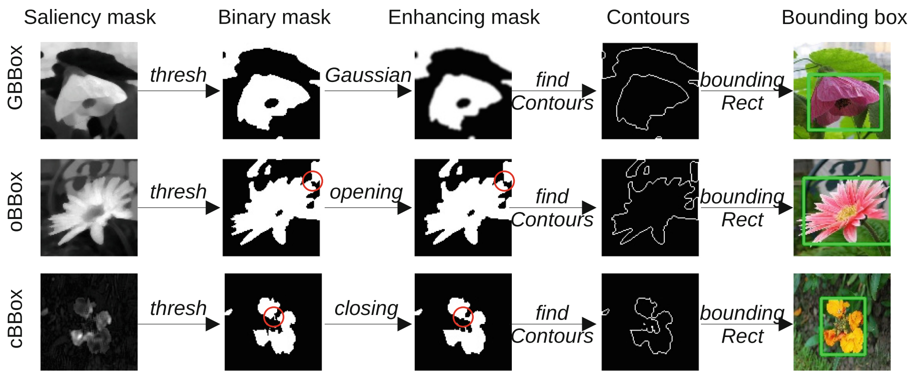

Fig. 2. The pipeline of the *GBBox*, *oBBox*, and *cBBox* functions.

The object localization process starts with the thresholding operation, which converts the saliency mask into a binary mask using a dynamic threshold method [13]. The main differences between *GBBox*, *oBBox*, and *cBBox* functions arise from their approaches to enhancing the binary mask based on different scene characteristics.

1) The opening operation in the*oBBox* function serves to eliminate noise, smooth object boundaries, and separate closely spaced objects.
2) The closing operation in the *cBBox* function aims to fill gaps or holes within objects, connect fragmented structures, and create more compact and complete object representations.
3) The *GBBox* function applies to the case where binary mask quality is satisfied and performs a 5×5 Gaussian filtering operation to smooth the binary mask.

Following the enhancement of the binary mask, the findContours operation identifies potential contours within this mask. The largest contour is considered the salient flower. Finally, the boundingRect operation calculates the bounding box of the largest contour as $[x, y, w, h]$, where $[x, y]$ represents the top-left point of the bounding box, and w and h indicate the width and height of the bounding box, respectively. The output of these object localization functions is represented as $[x, y, x+w, y+h]$, where $[x+w, y+h]$ denotes the bottom-right point of the bounding box. It's worth noting that, unlike existing GP methods, the calculated bounding boxes are different in each image, as they are based on the specific content of the image.

3.4 Fitness Function

The fitness function for ACFGP is the numerical sum of two metrics: detection accuracy and intersection over union (IoU). The formulas for these two metrics are given below.

$$\text{Fitness} = \text{Detection accuracy} + \text{Average IoU} \tag{1}$$

$$\text{Detection accuracy} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Average IoU} = \frac{1}{N}\sum_{i}^{N} \text{IoU}\left(\text{Prediction}_i, \text{Groundtruth}_i\right) \tag{3}$$

A bounding box is considered a true positive if it captures more than 50% (including 50%) of the target in terms of IoU, while a bounding box with less than 50% IoU is considered a false positive. However, detection accuracy alone cannot differentiate between cases with 50% and 100% IoU. To provide a more comprehensive evaluation for GP programs with the same detection accuracy, the fitness function also incorporates the average of IoU, where N denotes the number of training instances. As a result, the fitness function of ACFGP is a value ranging between 0% and 200%.

3.5 Test Process

In the test process, the best GP tree/program obtained from the evolutionary training process will be used to predict the images in the test set. According to the best GP tree, a set of aggregate channel features will be extended and aggregated to generate a saliency mask, and the bounding box of the salient flower will be calculated in each test image. Similar to fitness evaluation, the detection accuracy and average IoU will be calculated to evaluate the performance of the ACFGP approach but the results are shown separately.

4 Experiment Design

4.1 Datasets

The proposed ACFGP approach is developed for flower localization, which could be applied as image pre-processing for further fine-grained image classification (FGIC). Therefore, the performance of ACFGP is examined on the FGIC dataset: the Oxford_102 dataset. The Oxford_102 dataset contains 102 categories of flower images with 40–258 images per category. The images from the Oxford_102 dataset vary in size, and to ensure uniformity, all images are resized to 224 × 224 for input, which is also a commonly used size in training and testing.

To assess ACFGP's ability to flexibly extend and aggregate different channel features based on the characteristics of different flower species, experiments are conducted on five flower sub-datasets. Each sub-dataset consists of five classes randomly selected from the original Oxford_102 dataset, namely s1-s5 of flower5. To explore ACFGP's performance with limited training data, 10 images from each class are randomly selected as the training set, resulting in a training set of 10 × 5 images. The remaining images (ranging from 30 to 248 images per class) form the test set for each sub-dataset. For instance, to build flower5-s1, five flower species are randomly selected from the Oxford_102 dataset, with 10 images per class (50 images in total) used for training and the remaining images (279 images in total) assigned as the test set. This process is repeated five times to establish the five sub-datasets. Figure 3 presents example images of the sub-datasets used in the experiments. The flowers in the images vary in color and size, and exist in diverse backgrounds and lighting conditions. Additionally, the positions of the flowers differ in each image.

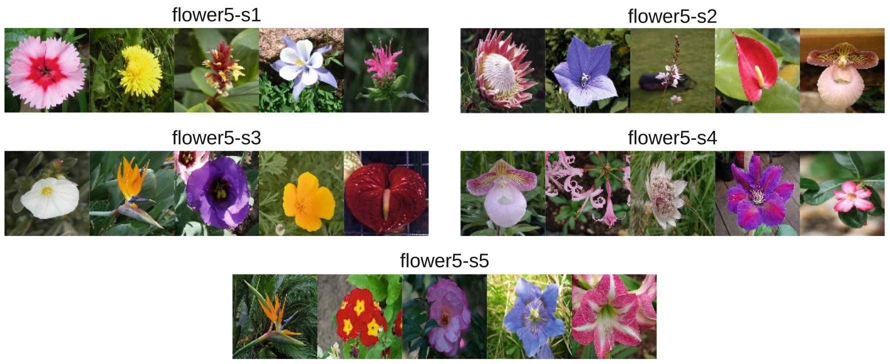

Fig. 3. Example training images of s1-s5 in flower5.

4.2 Comparison Methods

To assess ACFGP's performance, we compare it with Baseline and YOLOv8. Baseline serves as a benchmark for evaluating aggregate channel features against single-channel features in flower localization. YOLOv8 is chosen as a well-known and latest CNN method in object detection.

Baseline: The Baseline benchmark algorithm employs an exhaustive search across all single-channel features, representing possible programs generated by ACFGP's function set without channel aggregation functions. The search space comprises 171 possible solutions, which include 168 single-channel features extracted by 8 image filters and edge detectors (7 color channels × 8 channel extension functions × 3 object localization functions) and 3 feature maps generated by the *Saliency* function (with 3 object localization functions). For each sub-dataset, Baseline explores all possible solutions, and the program with the highest fitness value on the training set is selected as the best individual. Given its exhaustive nature, only a single-run experiment is conducted on each sub-dataset.

YOLOv8: The YOLOv8 method is a well-known and latest object detection model designed for identifying and localizing objects in images. Due to limited training data, the YOLOv8s (small) model from the YOLOv8 series [7] is used as a comparison method in this paper, which contains 11.2 million parameters.

4.3 Parameter Settings

ACFGP: The population size is 100 and the maximal number of generations is 50. The crossover rate is 0.5, the mutation rate is 0.49, and the elitism rate is 0.01. The Ramped-half-and-half method is used in population initialization and the mutation operation. The minimal and maximal tree depth are set to 2 and 6, respectively. Tournament selection with the size 5 is employed for selection. The ACFGP method is implemented using the DEAP package [4]. 30 independent runs of experiments are conducted with different random seeds for ACFGP on each sub-dataset.

YOLOv8: The YOLOv8s model pre-trained on the COCO dataset [9] is used for fine-tuning for comparison in this paper. The image size input is set to 224×224, the training epoch is set to 100, the batch size is set to 16, the optimizer is SGD, the learning rate is set to 0.01, and the remaining parameters follow the default setting in [7]. Ten independent experimental runs are performed with different random seeds for YOLOv8 on each sub-dataset. In the testing stage, considering that the YOLOv8 method permits multiple bounding box outputs within an image, the final output that localizes the flower is determined by selecting the bounding box with the highest confidence among multiple positive predictions.

5 Results and Discussions

This section compares the object localization performance, including the average IoU and the detection accuracy, of ACFGP, YOLOv8, and Baseline methods over the 30/10/1 runs on the s1-s5 of flower5. The evaluation focuses on two metrics: average IoU, which assesses the quality of object localization, and detection accuracy, which emphasizes the correctness of detections. The testing results, including the maximal (Max) and mean with standard deviation (Mean ± Std) for each metric, are presented in Tables 3 and 4, with the best results highlighted in bold. The Wilcoxon rank-sum test is used for the significance test with the *p-value* = 0.05. The symbols "+" or "–" in the tables indicate that ACFGP achieves significantly better or worse performance than the compared method. The symbol "=" indicates similar performance between ACFGP and the compared method. The final row in each table summarizes the significance test results on each sub-dataset.

5.1 Average IoU Results

Compared with Baseline: Table 3 demonstrates ACFGP consistently outperforming the Baseline across all comparisons, highlighting its superior ability to extend and aggregate more effective channel features for precise flower localization. This could be attributed to ACFGP's aggregate channel features, which capture diverse and complementary information, enhancing the discrimination of flower features.

Compared with YOLOv8: Table 3 illustrates that ACFGP significantly outperforms YOLOv8 on 3 out of 5 sub-datasets. While YOLOv8 achieves a higher maximal average IoU on flower5-s5, ACFGP maintains competitive performance overall on this sub-dataset. This suggests that ACFGP excels in localizing flowers with limited training data in most comparisons. It's worth noting that in 4 out of 5 comparisons, the standard deviation values of YOLOv8's results are larger than those of ACFGP, indicating that ACFGP achieves a more stable performance than YOLOv8. Despite being pre-trained, YOLOv8 usually demands thousands of flower images to fine-tune its model due to the high model complexity to achieve stable and satisfactory performance. Given the limited training data in this task, capturing discriminative features in flowers becomes challenging for YOLOv8, so it is reasonable that YOLOv8 shows inferior flower localization performance compared to ACFGP.

Table 3. Average IoU (%) of ACFGP, YOLOv8, and Baseline on flower5 sub-datasets

		ACFGP		YOLOv8		Baseline
		Max	Mean ± Std	Max	Mean ± Std	Max/Mean
flower5	s1	**79.12**	**77.53 ± 1.53**	78.97	73.71 ± 3.33+	75.65+
	s2	**81.66**	**77.20 ± 2.96**	79.62	75.35 ± 3.02=	63.88+
	s3	**82.57**	**80.59 ± 0.85**	77.30	73.01 ± 4.95+	78.84+
	s4	**84.03**	**80.90 ± 1.96**	82.22	79.23 ± 1.86+	78.71+
	s5	75.57	72.33 ± 1.79	**77.30**	**73.95 ± 3.18=**	72.21+
Overall					3+, 2=	5+

5.2 Detection Accuracy Results

Compared with Baseline: Table 4 reveals that ACFGP outperforms Baseline significantly in all comparisons. This suggests that ACFGP's aggregate channel features excel in localizing more objects with IoU over 0.5 compared to the single-channel feature in Baseline. Different channel features could highlight different aspects of the flower, such as the shape, texture, or color. Aggregating these channel features improves robustness against variations in lighting conditions and image noise. Consequently, ACFGP achieves better localization results with IoU over 0.5.

Compared with YOLOv8: As shown in Table 4, although YOLOv8 achieves better maximal detection accuracy on flower5-s1 and flower5-s5, ACFGP significantly outperforms YOLOv8 on 4 out of 5 flower sub-datasets in significance tests, and achieves competitive performance on flower5-s5. This indicates that ACFGP outperforms YOLOv8 in localizing more flowers with IoU over 0.5 in most comparisons, aligning with the performance observed in average IoU in Table 3.

Table 4. Detection accuracy (%) of ACFGP, YOLOv8, and Baseline on flower5 sub-datasets

		ACFGP		YOLOv8		Baseline
		Max	Mean ± Std	Max	Mean ± Std	Max/Mean
flower5	s1	89.42	**87.06 ± 1.99**	**89.61**	81.94 ± 4.21+	84.67+
	s2	**91.74**	**88.20 ± 2.58**	88.26	84.00 ± 4.08+	74.35+
	s3	**92.50**	**90.97 ± 0.98**	88.03	80.56 ± 4.76+	89.79+
	s4	**94.50**	**91.88 ± 2.08**	92.98	89.77 ± 1.85+	90.64+
	s5	87.06	82.77 ± 2.21	**88.10**	**82.83 ± 3.37=**	81.03+
Overall					4+, 1=	5+

6 Further Analysis

In this section, we analyze a GP tree/program evolved by ACFGP and compare the object localization performance between ACFGP, Baseline, and YOLOv8 to provide more insight into the effectiveness of the aggregate channel features.

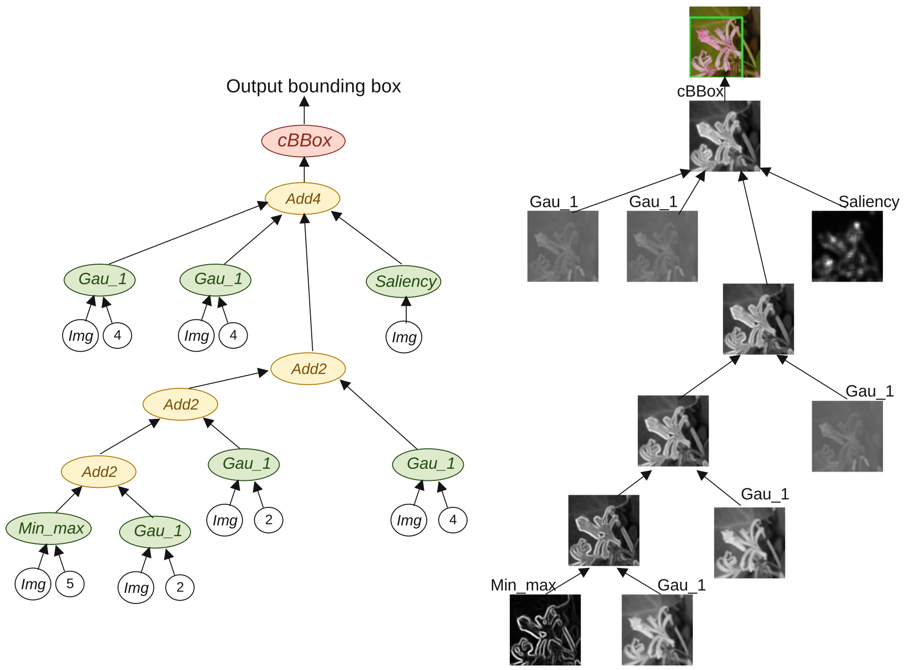

Fig. 4. An example program evolved by ACFGP on flower5-s4 and the example images to show the object localization process using the program.

6.1 Analysis of an Example GP Tree

Figure 4 presents an example GP tree evolved by ACFGP on flower5-s4, which achieves 82.47% average IoU and 93.13% detection accuracy. The program extends and aggregates seven channel features derived from three color channels (U, V, and red channels), incorporating 5 Gaussian features, 1 Min_max feature, and 1 Saliency feature. The corresponding example images reveal that the Gaussian features effectively suppress background noise, and the Min_max and Saliency features which contain edge information of the petal and visually important areas highlight the flower. The generated saliency mask plays a crucial role in reducing background noise and enhancing the flower, which results in a precise bounding box output.

The example GP tree showcases the effectiveness of the ACFGP approach, which autonomously selects informative color channels, extends them into effective channel features using diverse image filters and descriptors, and aggregates these features to generate the saliency mask for precise object localization.

6.2 Visual Comparison Between ACFGP, Baseline, and YOLOv8

Figure 5 displays the results of ACFGP, Baseline, and YOLOv8 across the five flower sub-datasets, alongside the saliency masks generated by ACFGP and Baseline for flower localization. These results are based on the best-run results of each method from the experiment.

Compared to the saliency mask created by the single-channel feature in Baseline, the saliency mask produced by the aggregate channel features in ACFGP exhibits greater robustness, leading to superior object localization performance across the five flower sub-datasets. While Baseline might accurately localize objects in certain instances using a single feature map (e.g., image *a* in flower5-s3), it struggles with challenges like shadow interference and complex backgrounds (as seen in images *c* and *d* in flower5-s3). In contrast, ACFGP's aggregated saliency mask integrates effective features from diverse perspectives, successfully mitigating background noise and shadow interference. It dynamically computes bounding boxes based on the image content, ensuring a robust and consistent object localization performance.

Due to the black-box nature of YOLOv8, detailed insights into its object localization process are unavailable, contributing to poor interpretability. When comparing results between ACFGP and YOLOv8, ACFGP shows better precision. In some instances, YOLOv8 fails to enclose the flower within blue bounding boxes (e.g., image *b* in flower5-s1, image *c* in flower5-s2, and image *b* in flower5-s4). Additionally, YOLOv8's results show notable background noise in several cases (e.g., image *e* in flower5-s1; image *d* and *e* in flower5-s4; and image *a*, *c*, and *d* in flower5-s5).

In summary, further analysis of the example GP tree/program evolved by ACFGP shows its high interpretability. The program provides a clear explanation of how the program extends and aggregates channel features to achieve precise flower localization. In addition, the comparison between ACFGP, Baseline, and YOLOv8 considering saliency masks and the corresponding results, highlights the effectiveness and robustness of aggregate channel features for precise flower localization.

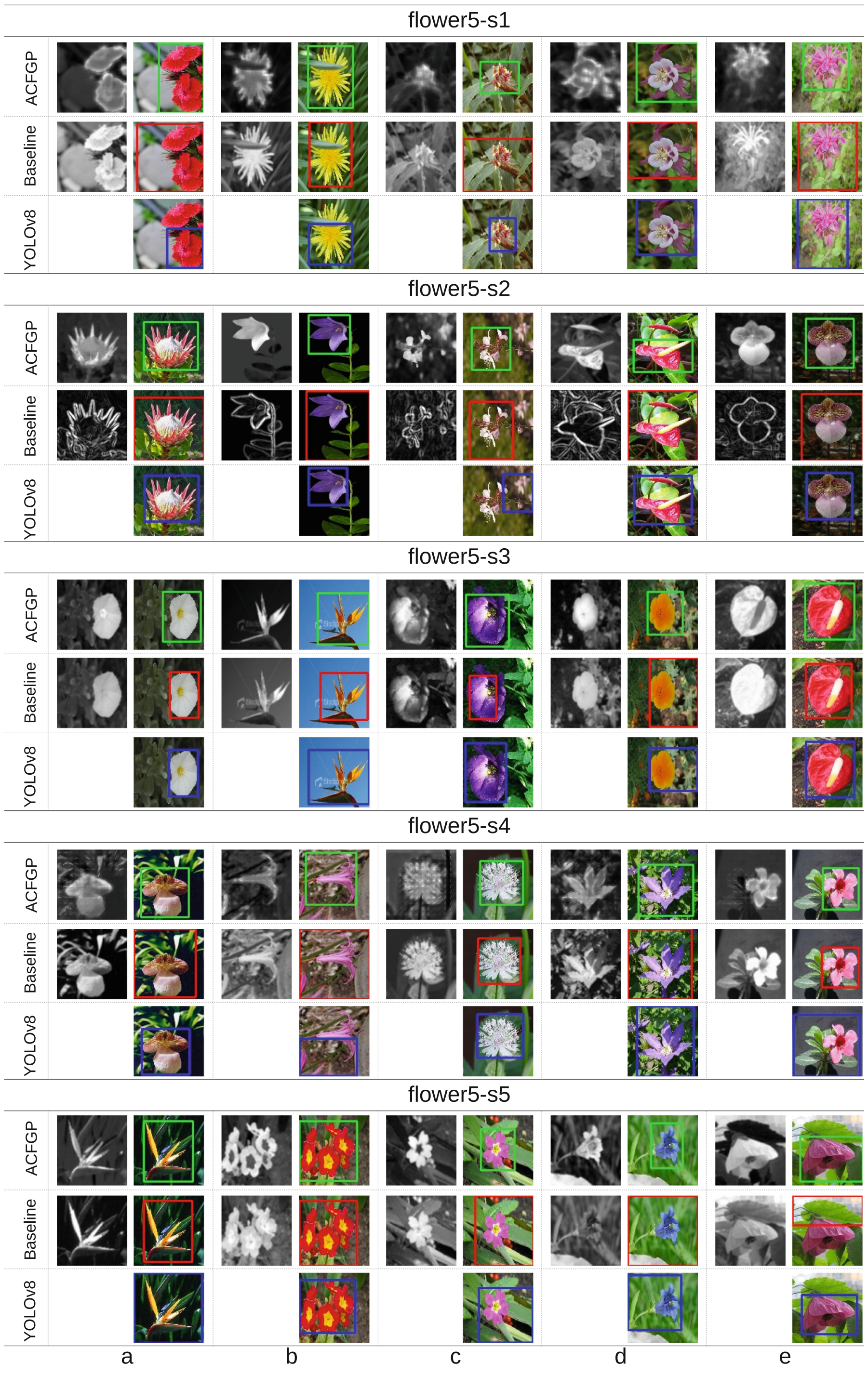

Fig. 5. The saliency masks and corresponding results of ACFGP, Baseline, and YOLOv8 on five flower sub-datasets.

7 Conclusions

The goal of this paper is to develop a GP-based approach to automatically extend and aggregate channel features for flower object localization. The goal has been successfully achieved by developing the ACFGP approach with a new GP representation, a new function set, and a new terminal set. This enables the simultaneous and automatic extension and aggregation of features, and dynamically localizing the flower in each image. The results show that the ACFGP outperforms the baseline benchmark and the YOLOv8 method. Further analysis highlighted the interpretability of the programs evolved by ACFGP, showcasing the effectiveness of the aggregate channel features for flower localization in ACFGP.

In the future, we will explore large-scale flower classification using the localization prediction results obtained through ACFGP.

References

1. Bi, Y., Xue, B., Mesejo, P., Cagnoni, S., Zhang, M.: A survey on evolutionary computation for computer vision and image analysis: past, present, and future trends. IEEE Trans. Evol. Comput. **27**(1), 5–25 (2023). https://doi.org/10.1109/TEVC.2022.3220747
2. Bi, Y., Xue, B., Zhang, M.: An effective feature learning approach using genetic programming with image descriptors for image classification [research frontier]. IEEE Comput. Intell. Mag. **15**(2), 65–77 (2020)
3. Dollár, P., Tu, Z., Perona, P., Belongie, S.: Integral channel features (2009)
4. Fortin, F.A., De Rainville, F.M., Gardner, M.A., Parizeau, M., Gagné, C.: DEAP: evolutionary algorithms made easy. J. Mach. Learn. Res. **13**(Jul), 2171–2175 (2012)
5. Guru, D.S., Sharath, Y.D.H., Manjunath, S.: Texture features and KNN in classification of flower images. Int. J. Comput. Appl. **1**, 21–29 (2010)
6. Jiang, P., Ergu, D., Liu, F., Cai, Y., Ma, B.: A review of YOLO algorithm developments. Procedia Comput. Sci. **199**, 1066–1073 (2022)
7. Jocher, G., Chaurasia, A., Qiu, J.: Ultralytics YOLO, January 2023. https://github.com/ultralytics/ultralytics
8. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
9. Lin, T.Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision-ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014, Proceedings, Part V 13, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
10. Montana, D.J.: Strongly typed genetic programming. Evol. Comput. **3**(2), 199–230 (1995)
11. Nilsback, M.E., Zisserman, A.: Delving into the whorl of flower segmentation. In: BMVC, vol. 2007, pp. 1–10 (2007)
12. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, pp. 722–729. IEEE (2008)

13. Ostu, N.: A threshold selection method from gray-histogram. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
14. Pardee, W., Yusungnern, P., Sripian, P.: Flower identification system by image processing. In: 3rd International Conference on Creative Technology CRETECH, vol. 1, pp. 1–4 (2015)
15. Patel, I., Patel, S.: An optimized deep learning model for flower classification using NAS-FPN and faster R-CNN. Int. J. Sci. Technol. Res. **9**(03), 5308–5318 (2020)
16. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
17. Rother, C., Kolmogorov, V., Blake, A.: " GrabCut" interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. (TOG) **23**(3), 309–314 (2004)
18. Siraj, F., Salahuddin, M.A., Yusof, S.A.M.: Digital image classification for Malaysian blooming flower. In: 2010 Second International Conference on Computational Intelligence, Modelling and Simulation, pp. 33–38. IEEE (2010)
19. Sun, Q., Yang, S., Sun, C., Yang, W.: Exploiting aggregate channel features for urine sediment detection. Multimedia Tools Appl. **78**, 23883–23895 (2019)
20. Wu, D., Lv, S., Jiang, M., Song, H.: Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. Comput. Electron. Agric. **178**, 105742 (2020)
21. Yan, Z., Bi, Y., Xue, B., Zhang, M.: Automatically extracting features using genetic programming for low-quality fish image classification. In: Proceedings of 2021 IEEE Congress on Evolutionary Computation (CEC), pp. 2015–2022. IEEE (2021)
22. Yang, B., Yan, J., Lei, Z., Li, S.Z.: Aggregate channel features for multi-view face detection. In: IEEE International Joint Conference on Biometrics, pp. 1–8. IEEE (2014)