scientific reports

OPEN

Counting manatee aggregations using deep neural networks and Anisotropic Gaussian Kernel

Zhiqiang Wang, Yiran Pang, Cihan Ulus & Xingquan Zhu[✉]

Manatees are aquatic mammals with voracious appetites. They rely on sea grass as the main food source, and often spend up to eight hours a day grazing. They move slow and frequently stay in groups (i.e. aggregations) in shallow water to search for food, making them vulnerable to environment change and other risks. Accurate counting manatee aggregations within a region is not only biologically meaningful in observing their habit, but also crucial for designing safety rules for boaters, divers, etc., as well as scheduling nursing, intervention, and other plans. In this paper, we propose a deep learning based crowd counting approach to automatically count number of manatees within a region, by using low quality images as input. Because manatees have unique shape and they often stay in shallow water in groups, water surface reflection, occlusion, camouflage etc. making it difficult to accurately count manatee numbers. To address the challenges, we propose to use Anisotropic Gaussian Kernel (AGK), with tunable rotation and variances, to ensure that density functions can maximally capture shapes of individual manatees in different aggregations. After that, we apply AGK kernel to different types of deep neural networks primarily designed for crowd counting, including VGG, SANet, Congested Scene Recognition network (CSRNet), MARUNet etc. to learn manatee densities and calculate number of manatees in the scene. By using generic low quality images extracted from surveillance videos, our experiment results and comparison show that AGK kernel based manatee counting achieves minimum Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). The proposed method works particularly well for counting manatee aggregations in environments with complex background.

Recent advancements in Artificial Intelligence (AI) have allowed it to be used in a wide spectrum of fields[1]. Especially, AI applications in ecology to save species have gained more importance in the last decade with the increasing number of endangered animals. There are many ways to use computational methods to help save endangered species, such as detecting the presence of the species[2] and counting species to collect information about numbers and density etc. In this paper, we are proposing a deep neural network based method to help save manatees, an endangered species.

Human activities impact environment in numerous ways including deforestation, overpopulation, poaching, over-fishing, and climate change. These negative impacts directly affect the physical environment, posing risks and generating opportunities for wildlife. Sometime, this dispersion benefits species, mostly invasive species[3]. In most cases, however, human-assisted dispersion adversely affects the natural world. As a result, wildlife populations are declining at an alarming rate, and the extinction rates are now up to 100 times higher than the normal extinction rate[4].

Manatees are one of the wildlife species being affected by human-related threats[5]. There are four species of manatees in the world: Trichechus inunguis (Amazonian manatee), Trichechus pygmaeus (dwarf manatee), Trichechus senegalensis (West African manatee) and Trichechus manatus (West Indian manatee)[6]. Amazonian manatees and dwarf manatees populate in freshwater habitats. The dwarf manatees are contentious species only found in the river Aripuaña in Brasil and closely linked to the Amazonian manatee[7,8]. The West African manatees are distributed from Angola to Senegal[8]. The West Indian manatee prefers shallow coastal habitats such as rivers and estuaries. They can be found from Brazil to Florida and all the way around the Caribbean islands. The West Indian Manatee has two subspecies; Florida Manatee and Antillean Manatee, and both of them are considered endangered by IUCN (International Union for Conservation of Nature)[9].

Department of Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL 33431, USA. [✉]email: xzhu3@fau.edu

While some manatee species, such as the Florida Manatee, have a relatively limited geographical distribution, some manatee types inhabit a wide geographical range. As a result of this wide range, individual manatee migrations occur[10]. For example, during the 2021 winter, a manatee migration from Florida to the Mexican coast was observed[11]. Furthermore, some types of manatees, such as Antillean manatees, can live in a diverse range of ecosystems. In some cases, they inhabit clear saltwater in Belize and Mexico, visit rivers, reefs, and freshwater lagoons, whereas they live in salt or fresh water in Chiapas and Tabasco states with insufficient visibility[12,13]. Such diverse living habits and behaviors make it inherently difficult to track them.

Over the last decades, manatee populations have been continuously decreasing. Manatees tend to live as a group or individuals. Most of their populations exists as small isolates and are also low in density[14]. Furthermore, they are frequently scattered throughout huge bodies of water and display evasive behavior due to hunting pressure, making detection and counting challenging with existing approaches. Knowing the number of manatees and their gathering pattern in real-time is vital for understanding their population dynamics. The timeliness and accuracy of the count data upon which choices are made frequently determine the efficacy of management decision-making. In other words, improvements in counting techniques may portend better ecological results from management decisions. Meanwhile, manatees rely on sea grass as their primary food source. Because sea grass requires sunlight and shallow water to grow, manatees tend to stay in shallow water to search for food, making them vulnerable to the environment, e.g. they have very little room/time to move away from (avoid) oncoming boats, resulting in deadly collisions if the boat drivers are not aware that they are approaching manatees.

Using aerial survey data, counting estimates for manatees in southern Florida, USA, was developed, and environmental and temporal factors were discovered to impact distributions[15,16]. However, aerial surveys are time-consuming and costly, and the accuracy depends on factors such as observer bias, weather, and time of the day. Consequently, less time-consuming and less costly counting methods gain more importance in detecting the number of manatees. Furthermore, it is also crucial to have a method to provide a real-time count to allow ecologists to be aware of the threat early and act proactively to protect manatees.

Although many methods exist for counting[17], most of the existing counting methods[18,19, 20] are applied to crowds to count the number of people, due to their relevance to important applications such as urban planning and public safety. Fortunately, such advanced techniques in crowd counting can also be generalized to other fields such as wildlife counting, by taking specific characteristics of the objects into consideration.

In this paper, we propose to use crowd counting methods to count manatee aggregations. Our goal is to accurately estimate number of manatees in a specific region, using low quality images as input. Due to numerous factors, as we have elaborated above, manatee counting is a challenging task.

- Occlusion: Because manatees tend to live in herd, they frequently block each others when viewing from the surface. As a result, small manatees are likely to be partially or completely blocked from the view.
- Distributions and Distortion: Due to diverse living habits and behaviors, manatees often present in different population density, perspective distortions, and lightning conditions. Without sufficient training data for each type of scene, it is difficult for a model to obtain accurate results for counting.
- Reflections and Camouflage: Furthermore, water reflections tend to make manatees invisible in reflection areas, counting manatees from images captured from surface mounted camera is difficult.
- Background: Finally, the high similarity of appearance between manatees and some elements in the background, such as fishes, rocks, imposes additional challenge to manatee counting.

In order to address the above challenges and accurately estimate the density of the manatee, we propose a deep neural network based crowd counting method, which learn to estimate manatee density within an input image. Our method considers distortions caused by the perspective between the water space and the image plane. Furthermore, since the shape of the manatee is closer to an ellipse than a circle, we propose a method that uses an Anisotropic Gaussian kernel (AGK) to best represent the manatee contour, and estimate manatee density in the scene. By formatting manatee counting as a deep neural network density estimation learning task, our approach balances the labeling costs vs. counting efficiency. As a result, our method delivers a simple and high throughput solution for manatee counting requiring very little labeling efforts.

Contribution

Our research brings the following three unique contributions to enrich our data and algorithm design for domain specific tasks:

- Deep Learning for Counting Manatees: We are among the first to introduce deep learning method to automatically count manatee through low-resolution images captured from surface mounted camera. This pioneering study not only addresses the technical challenges of counting in complex outdoor environments but also offers potential ways to aid endangered species.
- Anisotropic Gaussian Kernel with Line Label Annotation: We introduce Anisotropic Gaussian Kernel combined with line label annotations to generate density map. This novel method can represent the unique shapes of manatees and deliver more precise counting results.
- Manatee Counting Dataset: To validate our method and facilitate further research in this domain, we have developed a comprehensive manatee counting dataset, published through GitHub for public access (https://github.com/yeyimilk/deep-learning-for-manatee-counting).

Related work

Our research is closely related to two research tasks: (1) learning to count number of objects within a scene; and (2) obtaining labels to support counting.

Counting methods

Given the above-mentioned specific significance of counting, an increasing number of scholars have attempted to address the counting issue. Existing methods in the field mainly fall into three groups: detection-based[21–23], regression-based, and density estimation based approaches including Convolutional Neural Network (CNN) based density estimation techniques[24]. It is worth noting that with the superb performance of deep learning methods, models based on CNN have largely dominated a variety of counting tasks.

Previous counting methods mainly focused on counting people, although some methods focused on cars, animals, cells etc. Early works[22,23, 25] focused on counting people, using detection-based approach. Detection-based methods usually detect a head or person by using a sliding-window-like detector to count the number of people[26]. Recently many object detection tools (YOLO[27], R-CNN[28] etc.) are developed for object detection in sparse scenes. Nevertheless, these approaches do not show good performance in congested scenes because they require extraction of low-level characteristics.

Crowd counting, especially in real complex environments, has received increasing attention in various domains. A recent survey[29] indicates that majority count-related studies predominantly center around yield estimation, phenotyping, livestock monitoring, and insect monitoring, which together constitute approximately 97% of the applications[29].

A modified version of the Inception-ResNet architecture was employed to count tomatoes and simulated synthetic images were also used to enhance accuracy, although the system struggles to count green fruits[30]. A novel rice plant counting network, termed RPNet[31], has four modules: feature encoder, attention block, initial density map generator, and attention map generator. Results indicate that RPNet outperforms MCNN, CSRNet, SANet, TasselNetV2, and FIDTM on certain high-resolution image datasets. Many detection-based and segmentation-based network have been proposed for counting fruits or plants, including those specific to blueberries, wheat spikes, panicles, pistachios and grapes clusters[29].

The application of deep learning models, particularly CNN combined with aerial imagery captured through UAV (Unmanned Aerial Vehicle)[32] has shown significant promise in detection and counting. Models such as NasNet, Xception, YOLOv4, and YOLOv5 have demonstrated high accuracy, even under challenging conditions. While these models succeed in detecting cattle in various conditions, challenges like occlusion and diverse cattle breeds remain areas for improvement[33–35].

Our method addresses the unique challenges posed by counting submerged animals in outdoor open water environment, especially when dealing with long-shape entities such as manatees, from overhead webcam low quality images.

Annotation and labeling for counting

Labeling is a critical step to support learning. In order to learn to estimate number of objects in a scene, it is necessary to provide supervision (i.e. label information) for the learning algorithms. Such label information varies from labeling the whole image to each single object in the image.

Total count annotation

Using total count as a label in counting tasks can reduce annotation costs. This type of annotation is often referred to as Weakly-Supervised. Earlier counting frameworks[36,37] mostly used this approach. Borstel et al.[38] proposed a weakly-supervised solution based on the Gaussian process for density estimation. The training samples are divided into multiple subregions in their work, but each region is still annotated with only count labels. Modern counting methods try to dig for more information. Yang et al.[39] used a soft-label sorting network to sort multiple images in the presence of only numerical labels. Then they used a regression network with a shared backbone to obtain the final number of people. MATT[40] and Sam et al.[41] introduced a small number of point-level annotations and trained mainly using total count annotations. Their work shows that only a small number of point annotation samples can improve counting accuracy. JCTNet[42] and TransCrowd[43] use performant Transformer structures to regress the overall number directly. Further, CrowdMLP[44] uses a more concise multilayer perceptron (MLP) as the overall architecture and introduces a self-supervised agent task to impose spatial cues implicitly. Although the total count annotation-based approach has the lowest annotation cost, due to the lack of spatial information, the average counting performance is lower than that of the density map-based approach.

Bounding box annotation

In order to provide accurate information of the object in the scene, bounding box annotations draw a minimum outer rectangle for each object appearing in the image. This allows underlying learning algorithms to learn properties of the objects. Nevertheless, drawing precise rectangles is difficult and labor-intensive, especially for images with a high density of objects. In addition, bounding Box-based annotation methods cannot provide accurate predictions in extreme situations, such as low resolution or severe occlusion.

Dot annotation

Dot annotation, one of the most commonly used techniques in recent years, marks each counted object as a dot, where the dot is usually located at the center of the object, e.g. the center of a person's head. As a result, dot

annotation creates a locality point map where the sum of the point map equals to the total number of objects, and the learning can be carried out using information from the dots.

Dot annotation tremendously reduces the labeling efforts, compared to others like bounding box annotation, and is also able to handle extreme cases like object occlusions. In reality, it is, however, difficult to train a neural network (NN) using only sparse points. Therefore, point annotations are usually transformed into density maps, using transformation process defined as follows.

$$\mathscr{D}_I = \sum_{x \in L \in I} \mathbf{K}_\sigma(x) \tag{1}$$

where $\mathbf{K}_\sigma(\cdot)$ is a kernel function (e.g. a Gaussian kernel) and $\mathscr{D}_I$ is the density map of image I (generated using kernel functions). L is the set of labeled dot/point locations in the input image I, and σ is the scale parameter of the 2-D Gaussian kernel. When integrated, the density map contains per-pixel density information of the scene, which results in the count of object in the image.

Intuitively, the transformation in Eq. (1) blurs each point annotation according to the scale parameter σ, and settings of σ represent different types of density mapping. The most basic way is to keep σ as a fixed value, and an existing work[45] has studied commonly used σ parameters.

One limitation of using a fixed σ value is that it prevents the density map from capturing perspective information of the counted objects. When the scales of the objects varies significantly in the image, a fixed σ value results in low accuracy. Alternatively, an adaptive kernel[19] uses adaptive σ values by taking average distance obtained from k-nearest neighbor algorithm into consideration. Intuitively, it produces a severely blurred density map in a highly dispersed regions of the target to accommodate the scale variation due to perspective. However, this approach relies on the uniform distribution of the counted objects. When discrete targets are present, the k-nearest algorithm becomes unreliable.

Modern approaches try to introduce additional information to produce more reliable adaptive kernels. Liu et al.[46], Shi et al.[47], Yan et al.[48], and Zhang et al.[19] use a perspective map to smooth the final density map so that larger counting targets close to the camera have larger and smoother Gaussian regions in an attempt to eliminate the error due to perspective distortion. However, perspective mapping is unavailable in manatee counting because the water is not on the same plane. Some work[49] uses depth camera to obtain relative depth of the target to the camera to estimate the size of the target. However, depth acquisition of underwater objects is still an open problem, and is uncommon (due to cost) in general surveillance/tracking systems.

Figure 1 shows an example of different labeling/annotation approaches on an image with 21 manatees in the scene. From left to right, the labeling costs and difficulty increase. The left most image is labelled with a number 21, which is the number of manatees in the scene (the label does not provide any additional information about manatees, such as locations, orientation etc.). Dot annotation marks each manatee with a circular point, while bounding-box annotation outlines each manatee with a rectangular box. For comparison, our proposed method employs line-segment annotation, using a single line-segment to label each manatee.

Proposed method

Recently, density maps have been commonly used for presenting crowd counting because they can represent the distribution of the crowd. In our research, we propose to use Anisographic Gaussian Kernel (AGK) based crowd counting approach for manatee counting. In the following, we will first introduce kernel density based counting, and then propose manatee customized crowd counting framework.

Base network

Following recent approaches, we perform counting based on the density estimation framework. The input of the framework is an image $I \in \mathbb{R}^{w \times h}$, represented as a $w \times h$ matrix where w and h denote image width and height respectively. The ground-truth density map $\mathscr{D}_I \in \mathbb{R}^{w \times h}$ is used to train deep neural networks (DNNs) by imposing normalized 2-D Gaussian at manatee locations provided by the annotations. The deep neural network is trained to predict the density map $\hat{\mathscr{D}}_I \in \mathbb{R}^{w \times h}$ for an image I, such that the network predicted density map output $\hat{\mathscr{D}}_I$ is sufficiently close to the ground-truth $\mathscr{D}_I$.

Kernel function

A kernel, commonly used in machine learning to perform classification and clustering, is a non-linear mapping of two vectors in a feature space, through the dot product of two vectors. Given two $d-$dimensional vectors
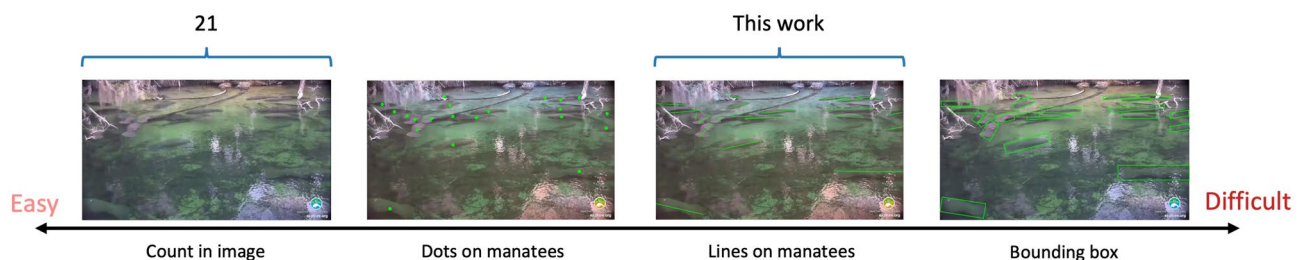

Figure 1. Comparison of different types of labels with their difficulties for the labeling work.

4

$\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$, and a transformation function $\varphi(\mathbf{x})$ defined for each vector, a kernel mapping between $\mathbf{x}_i$ and $\mathbf{x}_j$ is a function defined as dot product $\varphi(\mathbf{x}_i)$ and $\varphi(\mathbf{x}_j)$ as follows:

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j) \qquad (2)$$

For example, for 2−dimensional vectors $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^2$, with $\mathbf{x}_i = [x_{i,1}, x_{i,2}]$ and $\mathbf{x}_j = [x_{j,1}, x_{j,2}]$ a simple polynomial kernel is defined as

$$\begin{aligned} \mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) &= (1 + \mathbf{x}_i^T \mathbf{x}_j)^2 \\ &= (1 + x_{i,1}x_{j,1} + x_{i,2}x_{j,2})^2 \\ &= 1 + x_{i,1}^2 x_{j,1}^2 + x_{i,2}^2 x_{j,2}^2 \\ &\quad + 2x_{i,1}x_{j,1} + 2x_{i,2}x_{j,2} + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2} \\ &= \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j) \end{aligned} \qquad (3)$$

where $\varphi(\mathbf{x}_i) = [1, x_{i,1}^2, x_{i,2}^2, \sqrt{2}x_{i,1}, \sqrt{2}x_{i,2}, \sqrt{2}x_{i,1}x_{i,2}]$, and $\varphi(\mathbf{x}_j)$ is defined similarly.

The kernel function should satisfy the following three properties: symmetrical, non-negative, and the area under the curve of the function must be equal to 1.

There are some well-known examples of kernels satisfying specific properties, such as Gaussian kernel, multivariate Student kernel, and Laplacian kernel. The Gaussian kernel can be expressed as

$$\mathbf{K}_\sigma(\mathbf{x}_i, \mathbf{x}_j) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^d \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \qquad (4)$$

Similar to polynomial kernel, Gaussian kernel function in Eq. (4) can be transformed as dot product between two vectors, $\varphi(\mathbf{x}_i)$ and $\varphi(\mathbf{x}_j)$, by using Taylor series to expand the kernel function into an infinite series of products.

The original Gaussian kernel has the same spread (σ) for all feature dimensions. As a result, it is difficult to represent a high dimensional space and differentiate features more important to capture the decisions (or classifications)[50].

Kernel density map generation
Given an image $\mathbf{I} \in \mathbb{R}^{w \times h}$ represented as a $w \times h$ array, and a set of n labelled points $\mathbf{p}_1, \ldots, \mathbf{p}_n$ in the image (e.g. using dot annotation), the kernel density map intends to generate a density map $\mathscr{D}_I \in \mathbb{R}^{w \times h}$ of I, with respect to the labelled points L (so the density map is primarily focused on labelled points).

The kernel density estimator (KDE) is a non-parametric estimator used to estimate the univariate or multivariate densities based on kernels as weights[51]. One direct way to create a kernel density map $\mathscr{D}_I$ is to compare each pixel in I to each labeled point $\mathbf{p}_i \in \mathbf{R}^2$, using 2-D location to represent pixels and points.

Denote $\mathbf{x}_{i,j} \in \mathbb{R}^2$ the 2-D location of a pixel of image I located at $[i, j]$, kernel density map $\mathscr{D}_I$ can be calculated using kernel transformation below

$$\mathscr{D}_I[i,j] = \frac{1}{n} \sum_{k=1}^n \mathbf{K}_\sigma(\mathbf{x}_{i,j}, \mathbf{p}_k); \qquad (5)$$

where $i = 1, \ldots, w$; $j = 1, \ldots, h$ are indices of image width and height, and $\mathbf{K}_\sigma$ is a kernel function with bandwidth σ.

Density map to counting
After obtaining the density map of an image, the number of manatees can be calculated by using element-wise summation of all points' density value as follows

$$C_I = \sum_{i=1}^w \sum_{j=1}^h \mathscr{D}_I[i,j] \qquad (6)$$

where C_I denotes ground-truth manatee numbers in the image I. For each labeled image, the annotations have ground-truth C_I value, so density map needs to be normalized accordingly to ensure that the sum of element-wise density map equals to its C_I value.

Figure 2 shows an example of an input image I with dot annotations (left panel), and its density map $\mathscr{D}_I$ based on labeled points (right panel). The ground-truth C_I is 21, because there are 21 manatees within the image.

After obtaining the density map $\mathscr{D}_I$, we will train neural network using input image I, and using density map $\mathscr{D}_I$ as expected output of the network. A sufficiently trained neural network is therefore capable of learn to detect an input image's manatee locations as a density map. By calculating element-wise sum of the predicted density map, using Eq. (6), we can calculate number of manatees in the scene (detailed in the latter section).

Manatee customized crowd counting
Line-label for manatee counting
Although point label are commonly used for generating density map in crowd counting, a manatee's body is oval-shaped, meaning that point annotations cannot effectively capture orientation and shape of manatees for
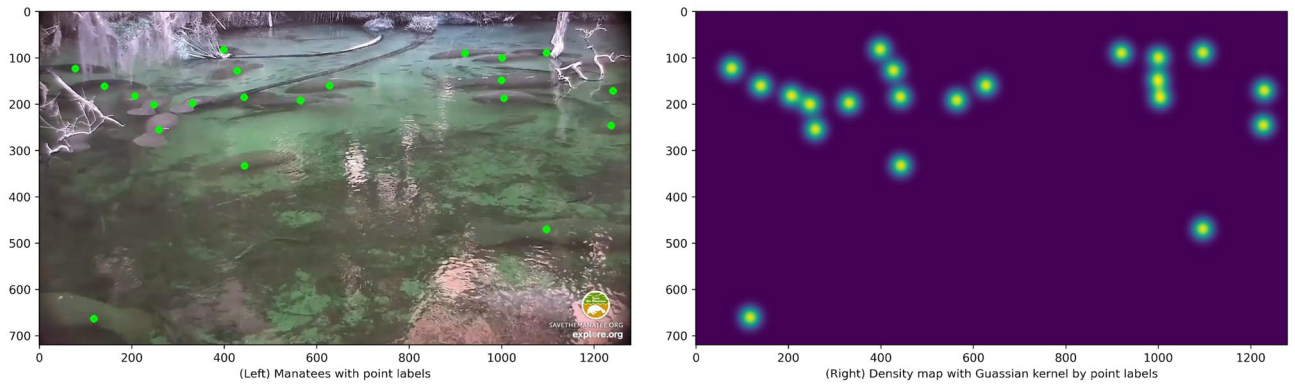
Figure 2. Left panel: an image with dot labels of manatees, and Right panel: density map of the image generated by applying Gaussian distributions to labeled points.

accurate counting. An alternative solutions is to use line-segment labels, e.g., a straight line segment, to mark each manatee.

A challenge of using a line-segment to annotate each manatee is that a line consists of an infinite number of points, making it difficult to obtain a density map of the input image. In this paper, we propose to use a limited number of points to represent the segment, and then use Gaussian kernel to denote each point for density map generation.

More specifically, given two endpoints, $\mathbf{x}_a = [x_{a,x}, x_{a,y}]$ and $\mathbf{x}_b = [x_{b,x}, x_{b,y}]$, of a line-segment, we generate $\lceil |x_{b,x} - x_{a,x}| + 1 \rceil$ number of Gaussian kernels using position evenly distributed on the line (where $\lceil \cdot \rceil$ denotes a ceiling function). In addition, to better capture manatee shapes, we generate an oval shaped Gaussian kernel for each dot, using σ value, which is adjusted according to the position of current point $\mathbf{x}_i = [x_{i,x}, x_{i,y}]$ comparing to the two endpoints of the line segment, as defined in Eq. (7), where a is a constant value (i.e. a parameter).

$$\sigma = \sigma_{basic} + a \cdot \min(\|\mathbf{x}_i - \mathbf{x}_a\|, \|\mathbf{x}_i - \mathbf{x}_b\|) \tag{7}$$

After obtaining σ value, we can use point $\mathbf{x}_i$'s current location as mean (μ) and generate a Gaussian distribution corresponding to point $\mathbf{x}_i$. Repeating this process, one can generate many Gaussian kernels for each line-segment. The total number of Gaussian kernels for each input image (which often contains many line segments) are used to normalize Gaussian kernels such that the density map of the whole image follows a distribution.

To illustrate the density map generation using line-segment, Fig. 3 shows examples of using single point and a line-segment to generate manatee customized density map. From left to right, Fig. 3(I) denotes a single point at (14, 14), while the subfigure (II) is the density map generated by given head coordination of the Gaussian kernel with $\sigma = 4$. Figure 3(III) is a line-segment from point (5, 5) to point (25, 25) which includes 21 points in total, and Fig. 3(IV) denotes the density map generated by the given points in Fig. 3(III) where the result has been normalized. In this subfigure, the smallest σ, $\sigma_{basic} = 3$, is 3 at the two endpoints, while the largest σ value is 5 corresponding to the Gaussian kernel at the center point of the line-segment at (15, 15).

The results in Fig. 3 show that using a line-segment, combined with Gaussian kernels, can provide a customized density map resembling a manatee's oval-shape for counting estimation.

Anisotropic Gaussian Kernel for manatee counting
While the oval-shaped density map in Fig. 3(IV) resembles to manatees' shape and orientation, it does not effectively capture the width of manatees' body, where the length of a manatee is much longer than its width. AGK kernel is a modification of the Gaussian kernel (4). Instead of using a single kernel parameter for all features, AGK kernel uses different kernel parameters to differentiate Gaussian distributions along each feature dimension. The AGK kernel function can be defined as
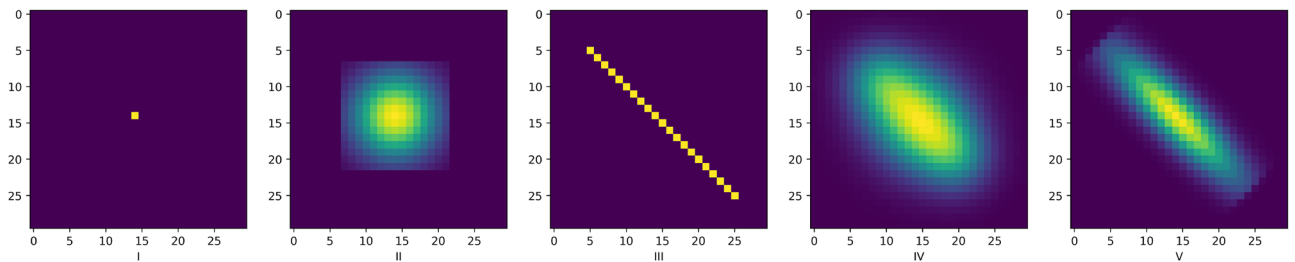

Figure 3. (I) A single point at (14, 14); (II) Gaussian with $\sigma = 4$; (III) line-segment from point (5, 5) to (25, 25); (IV) Gaussian with $\sigma_{basic} = 3$ and $a = 0.2$; (V) Anisotropic Gaussian with $\sigma_{x_1} = 8.24, \sigma_{x_2} = 2.06$ for line-segment in (III).

$$\mathbf{K}_\sigma\left(\mathbf{x}_i, \mathbf{x}_j\right) = \exp\left(-\sum_{k=1}^{d} \frac{\left\|x_{i,k} - x_{j,k}\right\|^2}{2\sigma_k^2}\right) \tag{8}$$

where $k = 1, \ldots, d$ denotes the feature indices. Because manatees' density maps are 2-D images, we have $d = 2$ for manatee counting. To generate a density map that most closely resembles each individual manatee, parameter σ_k is dynamically adjusted using Eq. (9).

$$\begin{cases} \sigma_1 = \frac{\|\mathbf{x}_a - \mathbf{x}_b\|}{2} \times \frac{\text{FWHM}}{\alpha} \\ \sigma_2 = \frac{\sigma_1}{A\!R} \end{cases} \tag{9}$$

in Eq. (9), $\|\mathbf{x}_a - \mathbf{x}_b\|$ denotes the length of the underlying line-segment. FWHM denotes Full Width at Half Maximum of the Gaussian distribution defined by the line-segment. It is used to adjust the σ to make the distribution more centralized. Intuitively, given a Gaussian distribution with σ standard deviation value, FWHM and σ satisfy following relationship: $\text{FWHM} = 2\sqrt{2 \ln 2}\sigma \approx 2.355\sigma$. To allow flexibility, we use a parameter α to penalize the FWHM, and control Gaussian distribution spread.

Overall, σ_1 is adjusted based on the length of the line-segment and the FWHM. For σ_2, it is also adjusted based on the σ_1 value and the empirical Aspect Ratio ($A\!R$), i.e. length divided by width, of the underlying object. In our experiments, we empirically set a fixed $A\!R$ value for all experiments.

Figure 3(V) shows an example of the density map generated for a line-segment between (5, 5) and (25, 25) by setting the AGK parameters with $\sigma_1 = 8.24$ and $\sigma_2 = 2.06$. Comparing Fig. 2(IV) and (V), i.e. density maps generated using simple Gaussian kernel vs. Anisotropic Gaussian kernel (AGK) respectively, the results show that AGK based density map is more resemblance to the manatee shape, and therefore will result in more accurate counting for manatees.

In Fig. 4, we report a manatee aggregation with line-segment labels (left panel), the density map generated from Gaussian kernels (middle), and the density map generated from AGK kernels, by adding oval shaped Gaussian distributions over the lines. Comparing two density maps, we can find that density map from AGK kernels provide more accurate density estimation with respect to individual manatee's shape and position. For example, the two manatees to the upper-right scene are represented as two circular-shaped areas, whereas AGK kernel's density shows clear oval-shapes with accurate orientation and aspect ratios.

CSRNet learning to predict density map

After obtaining density maps for each labeled images, our next step is to use original images to train a deep neural network. Once the network is sufficiently trained, for each input image, the network will output a density map maximally approximating to the ground-truth density map of the input image.

More specifically, given a labelled training deadset $\mathscr{I}$ with $|\mathscr{I}|$ images where each image I is labeled and has a ground-truth density map $\mathscr{D}_I$. Denote $f_\Theta()$ a deep neural network regulated by a number of tunable parameters Θ, and for each labeled input image $I_i \in \mathscr{I}$ and its density map $\mathscr{D}_{I_i} \in \mathscr{D}$, the network will output a predicted density map $\hat{\mathscr{D}}_{I_i}$, the neural network aims to learn optimal parameters Θ^*, such that total loss $\ell()$ with respect to the predicted density map is minimized.

$$\Theta^* = \arg\min_\Theta \sum_{I_i \in \mathscr{I}} \ell(f_\Theta(I), \mathscr{D}_{I_i}) \tag{10}$$

To learn to predict density map, we employ Congested Scene Recognition network (CSRNet) as our basic model. CSRNet is designed to understand highly congested scenes and perform accurate count estimation as well as produce high-quality density maps[45]. The CSRNet consists of two major components, including a CNN (convolutional neural network) as the front-end for 2-D feature extraction and a dilated CNN for the back-end to replace pooling operations. In our design, we use VGG16[52], a pretrained CNN network, as the front-end of CSRNet because it has strong transfer learning ability and flexibility to concatenate the back-end for density map generation. The fully-connected layers of VGG-16 is removed in the CSRNet.

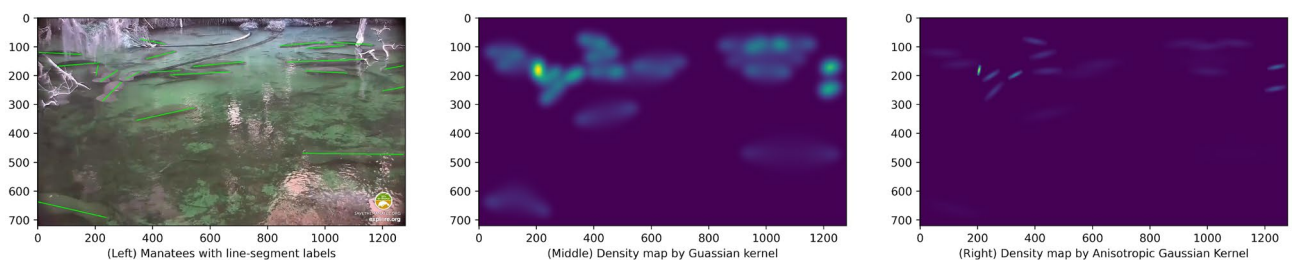For the back-end, a 2-D dilated convolution can be defined as

Figure 4. Left: an image with line labels of manatees, Middle: density map of the image generated by using generic Gaussian kernels (line-segment labels), and Right: density map generated by using AGK kernels (line-segment labels).

$$y(m, n) = \sum_{i=1}^{M} \sum_{j=1}^{N} x(m + r \times i, n + r \times j)w(i, j) \qquad (11)$$

where $y(m, n)$ is the output, $x(m + r \times i, n + r \times j)$ is the input, $w(i, j)$ is the filter with M length and N width respectively, and r is the dilation rate. When $r = 1$, the dilation convolution layer is the same as a normal convolution layer.

In the dilated convolution layer, a $k \times k$ kernel filter will be enlarged to $(k + (k - 1)(r - 1)) \times (k + (k - 1)(r - 1))$ with dilated stride r. Because of the application of the dilated convolution, the CSRNet can maintain the resolution of feature map. Most importantly, the output from dilated convolution contains more detailed information.

Overall framework

Algorithm 1 outlines major steps of the proposed framework for manatee counting using DNN and AGK kernels. In addition, Fig. 5 also shows conceptual flow of the framework with respect to input images, annotations, density maps, and DNN network training.

The general workflow can be splitted into two phrases, annotation and training phrase, and prediction phrase. During annotation and training phrase, all training images are labelled, and each original image and its labels are used to generate AGK density map (detailed in Algorithm 3). After this step, all training images and their density maps are used to train a deep neural network, by using Eq. (13) to calculate the loss between the DNN output and each training image ground-truth density map. The loss is used to update weights of the DNN until the network converges. After that, in the prediction phase, the trained neural network can be utilized to predict number of manatees in a test image.

Density map generation
One of the main steps in Algorithm 1 is to generate AGK density map (Algorithm 3) for each image I (this step can also be replaced by Algorithm 2 to generate normal Gaussian Kernel Density Map). For Gaussian distributions, the probability density values are continuous infinitely with respect to the input space. In reality, the distributions are restricted to a 2-D window, limited by the image size. From distribution perspective, for a generated Gaussian distribution, the sum all values within the 2-D array should be equal to 1. Therefore, we truncate each single Gaussian distribution to a specific region, and all distributions' output, and further normalize the values to ensure that they follow a distribution. In the following, we explain steps to generate three types of density map, dot-label, line-label, and line-label Anisotropic Gaussian, respectively.

Dot-label Gaussian density map. To obtain a dot-label Gaussian density map, n Gaussian distributions are generated based on the number of dots (i.e. labels) within the image. Then these n 2D arrays are added to their corresponding point positions in the map, which is an image-sized initialized 2D array without the color dimension.

Line-label Gaussian density map. To get a line-label Gaussian density map, its general algorithm is shown as Algorithm 2. Firstly, a w by h shape 2D array, $\mathcal{O}$, is initialized with 0 where w and h are the width and height of the Image I, respectively. Then get each l from labels, L, where the label contains two points, start point and end point of the line. A series of points is generated pixel by pixel from start point to end point. For each of the point, a σ value is created according to the Eq. (7) followed by creating its corresponding window size. After that, a Gaussian distribution array can be generated and added to a temporary density map o at its position. Once Gaussian distributions are generated for all of points of the line-segment, the o is normalized because one line label is only regarded as one count. Finally, the o is added back to the output $\mathcal{O}$.

Line-label anisotropic Gaussian density map. Algorithm 3 lists steps to generate a line-label anisotropic Gaussian density map. An array of $\mathcal{O}$ is initialized with 0 which is the same as it in generating a line-label Gaussian density map. For each of the label, l, in labels, L, first, the length, len, of the line-segment is calculated. With the length of the segment, σ_1 is calculated according to Eq. (9) followed by σ_2. Then, generating anisotropic Gaussian distribution based on σ_1 and σ_2 within a 2-D array, $\mathcal{N}(\mu, \sigma)$, with the shape of len by len. After that, it rotates the generated distribution respecting to the slope of the label. Before adding $\mathcal{N}^\circ(\mu, \sigma)$ back to the output $\mathcal{O}$ at the position of the center of the line-segment, $\mathcal{N}^\circ(\mu, \sigma)$ is normalized. Once all the distributions have been created for the labels, anisotropic Gaussian density map is presented for the image I.

Figure 5. A conceptual view of the deep neural network based manatee counting workflow. From left to middle: An input image (I), the line-segment labels of the image, the density map $\mathcal{D}_I$ of the image. From right to middle: An image I is applied to a neural network (NN) to predict its density map $\widehat{\mathcal{D}}_I$. The loss function in the middle is used to regulate and train the NN to ensure that $\widehat{\mathcal{D}}_I$ maximally approximates to the ground-truth $\mathcal{D}_I$.

```
    Data: (1) 𝓘: training image set;
          (2) 𝓣: test image set;
 1  Define: (1) 𝓓: density map set;
 2          (2) f_Θ(): a DNN, e.g. a CSRNet or other DNNs;
 3          (3) a(I): label an image I and return annotations;
 4  === Annotation and training phase ===
 5  𝓓 ← {}
 6  for I ∈ 𝓘 do
 7  │   L ← a(I) ;                                  // obtaining training image I annotations
 8  │   D ← AGKDensity(I, L) ;                              // obtaining I's AGK density map
 9  │   𝓓 ← 𝓓 ∪ D
10  end
11  f_Θ*() ← arg min ℓ(f_Θ(𝓘), 𝓓) ;                    // training DNN network using Eq. (10)
           Θ
12
13  === Prediction phase ===
14  for T ∈ 𝓣 do
15  │   𝓓̂^T ← f_Θ*(T) ;                                      // predicting density map
16  │   C^T ← Calculate manatee numbers in T using Eq. (6)
17  end
```

Algorithm 1. DNNs for Manatee counting using AGK

Limitations

The proposed anisotropic Gaussian kernel relies Gaussian distributions to generate a density map to approximate manatee shapes. By doing so, it relies on assumption that manatee shapes are largely visible and are close to the AGK kernel shape. As a result, several potential limitations may hinder this approach from obtaining high accuracy counting results:

- Close-up Views: When manatees are situated too close to the camera, they may appear disproportionately large, with only parts of their bodies being captured. In such cases, line-labels fail to accurately mirror the shape of the manatee. The resultant density graph might not correspond well to the actual manatee shape, and result in poor counting performance.
- Distant Views: On the opposite end, manatees that are extremely further away from the camera appear diminutive, resembling dots. Under these conditions, the differentiation between line-label and dot-label becomes negligible, resulting in comparable accuracy for both annotation methods.

These real-world scenarios outline the importance of understanding the inherent constraints of different labeling/ annotation approaches for real-world applications. Our experiments and comparisons demonstrate the strength and niche of AGK kernels in counting manatee under different environment conditions.

Experiments
Benchmark data

In order to validate the performance of proposed framework using low resolution images, manatee surveillance video clips from "Save the manatee Club" (https://www.savethemanatee.org/manatees/manatee-webcams/), which are captured from webcams placed at Blue Spring State Park, are collected to create our benchmark dataset. After using FFmpeg to generate images from the video clips and removing similar images, we obtain 784 images, consisting of different number of manatees, as our testbed.

Deduplication

Because manatee images are collected from video clips and manatees rarely show rapid movement, some of the images are similar to each other. To avoid data duplication, duplicate images are dropped by using VGG-16 to extract layer-wise features for each image and then a method is employed to calculate the difference between two images, I_a and I_b, with the formula being defined as follows:

$$\text{Disance}(I_a, I_b) = \sum_{j \in \mathbb{F}} \frac{1}{C_j H_j W_j} \left| \left| \psi_j(I_a) - \psi_j(I_b) \right| \right|_2^2 \tag{12}$$

$\mathbb{F}$ is the outputs from the last activation layer in each group of VGG-16's 5 groups, namely $\mathbb{F} = \{\psi_j | j = 1, 2, 3, 4, 5\} = \{layer_2, layer_4, layer_7, layer_{10}, layer_{13}\}$. C_j, H_j, W_j are the three dimensions of the image, channel, height, and width, respectively. $\psi_j(i_x)$ denotes the jth group ReLu output of the image i_x. The smaller value of Distance(I_a, I_b), the more similarity of the two images is considered. When the value of Distance(I_a, I_b) is less than 2, the two images are considered highly similar, then one of the two images is dropped.

After comparing similarities of the images, 415 of 784 images are discarded and only 369 images are kept as dataset.

Data: (1) I: an input image;
  (2) L: line labels of image I
1 **Define:** (1) $\mathbf{l}_y$ and $\mathbf{l}_s$: two endpoints of line segment $\mathbf{l}$
2   (2) $\mathscr{N}(\mu,\sigma)$: a multidimensional Gaussian distribution with μ mean and σ spreads
3 **Input:** (1) a: expanding factor;
4   (2) σ_{basic}: basic value for generating σ;
5 **Output:** $\mathscr{O}$: density map of image I
6 $[w,h] \leftarrow$ Width and height of image I $\mathscr{O} \in \mathbb{R}^{w \times h} \leftarrow \mathbf{0}$;   `// create an empty density map`
7 **for** $\mathbf{l} \in L$ **do**
8   $[\mathbf{x}_a, \mathbf{x}_b] \leftarrow [\mathbf{l}_s, \mathbf{l}_e]$;   `// two ends of a line-segment`
9   $\mathbf{P} \leftarrow$ Evenly select $\lceil |\mathbf{x}_{a.x} - \mathbf{x}_{b.x}| + 1 \rceil$ points from line-segment $\mathbf{l}$
10   $\circ \in \mathbb{R}^{w \times h} \leftarrow \mathbf{0}$;   `// temporary density map`
11   **for** $\mathbf{p} \in \mathbf{P}$ **do**
12     $\mu \leftarrow \mathbf{p}$ `// Gaussian distribution mean`
13     $\sigma \leftarrow \sigma_{basic} + a \cdot \min(\|\mathbf{p} - \mathbf{x}_a\|, \|\mathbf{p} - \mathbf{x}_b\|)$;   `// Gaussian distribution spread Eq. (7)`
14     $\mathscr{N}(\mu,\sigma) \leftarrow$ Generate 2D Gaussian Distribution
15     $\circ \leftarrow \circ + \mathscr{N}(\mu,\sigma)$;   `// add current Gaussian distribution to the temporary density map`
16   **end**
17   Normalize temporary density map $\circ$
18   $\mathscr{O} \leftarrow \mathscr{O} + \circ$
19 **end**
20 **return** $\mathscr{O}$

Algorithm 2. LineDensity()—Gaussian Kernel density map generation via line labels

---

Data: (1) I: an input image;
  (2) L: line labels of image I
1 **Define:** (1) $\mathbf{l}_y$ and $\mathbf{l}_s$: two endpoints of line segment $\mathbf{l}$
2   (2) $\mathscr{N}(\mu,\sigma)$: a multidimensional Gaussian distribution with μ mean and σ spreads
3 **Output:** $\mathscr{O}$: density map of image I. $[w,h] \leftarrow$ Width and height of image I
4 $\mathscr{O} \in \mathbb{R}^{w \times h} \leftarrow \mathbf{0}$;   `// create an empty density map`
5 **for** $\mathbf{l} \in L$ **do**
6   $[\mathbf{x}_a, \mathbf{x}_b] \leftarrow [\mathbf{l}_s, \mathbf{l}_e]$;   `// two ends of a line-segment`
7   $\mu = [\mu_1, \mu_2] \leftarrow \frac{\mathbf{x}_a + \mathbf{x}_b}{2}$;   `// calculate AGK mean`
8   $\sigma = [\sigma_1, \sigma_2] \leftarrow$ Eq. (9) ;   `// calculate AGK spreads`
9   $\mathscr{N}(\mu,\sigma) \leftarrow$ Generate 2-D Gaussian Distribution
10   $\circ \leftarrow$ Calculate slope between $\mathbf{x}_a$ and $\mathbf{x}_b$
11   $\mathscr{N}^\circ(\mu,\sigma) \leftarrow$ Align 2-D Gaussian Distribution $\mathscr{N}(\mu,\sigma)$ based on the line-segment slope
12   Normalize $\mathscr{N}^\circ(\mu,\sigma)$
13   $\mathscr{O} \leftarrow \mathscr{O} + \mathscr{N}^\circ(\mu,\sigma)$ `// add current Gaussian distribution to the density map`
14 **end**
15 **return** $\mathscr{O}$

Algorithm 3. AGKDensity()—AGK density map generation via line labels

Dataset characteristics
All 369 benchmark images have the same size. According to the number of manatees in each image, we separate each image into three density levels: low, medium, and high. We define that, if an image has less than 5 manatees, then it is of low density level. Continue with medium level if it has more or equal to 5 but less than 20 manatees. If an image with more than 19 manatees, it is of high density level.

Figure 6 reports the manatee density distributions in the benchmark images at three levels. Images with low density level have the highest quantity (181), which is almost equal to the total amount of medium and high level which have 84 and 104 images, respectively.

Experimental settings
Manatee annotations
To validate the performance of using Anisotropic Gaussian Kernel (AGK) for manatee counting, we use dot and line labels to annotate images for performance comparison. In order to reduce duplicate manual label work, we draw a line over each manatee (from tail to head) and make sure the line crosses over the center point of the manatee (the number of labelled line segments within an image is the number of manatees in the image). Consequently, two endpoints of a line for each manatee are obtained and saved into a JSON file for further usage. Assuming $\mathbf{x}_a = [x_{a,x}, x_{a,y}]$ and $\mathbf{x}_b = [x_{b,x}, x_{b,y}]$ are the two endpoints of a labelled line segment obtained from previous stage, $[\frac{x_{a,x} + x_{b,x}}{2}, \frac{x_{a,y} + x_{b,y}}{2}]$ is used as point label for the same manatee.

Parameter settings
We validate the performance of multiple baseline methods on our manatee dataset over three labels. Compared to traditional point labeling methods[45], our line labeling adds only a small amount of additional time and obtains
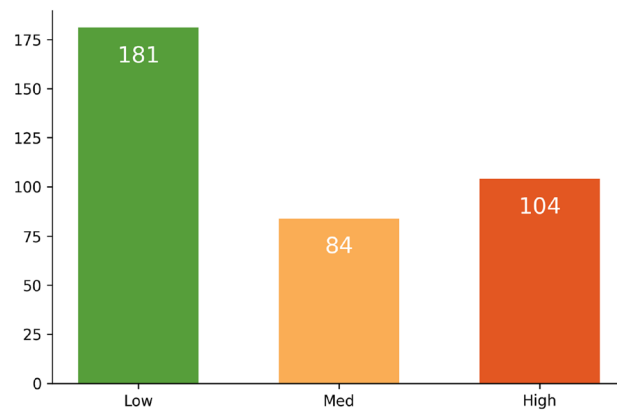
Figure 6. Distribution of benchmark images w.r.t. different manatee density levels. The *x*-axis denotes manatee density levels and the *y*-axis denotes number of images.

an overall better performance. In this section, we report the evaluation metrics and perform an ablation study on the three labeled datasets to study their performance.

We evaluate the object counting task on an NVIDIA V100 GPU card. The watermark on the surveillance video images at the lower right corner is filled with solid black to avoid additional interference. During training, the initial learning rate is 1e-4, and the Adam optimizer is used. For better training and to prevent overfitting, random flips were used for augmentation. For all networks, the batch size is set to 4.

The network directly converts the input image into a density map in training. During the density map generation, some hyperparameters are used in Algorithms 2 and 3 . They are the expanding factor, $a = 0.2$, the base value of σ, $\sigma_{basic} = 15$, the empirical Aspect Ratio, $Æ\!R = 4$, full width at half maximum FWHM $= 2.355 \approx 2\sqrt{2\ln 2}$, and the FWHM penalizer α in Eq. (9) is set as $\alpha = 4$. Intuitively, the FWHM and α parameters in Eq. (9) are determined such that the σ_1 value is roughly $\frac{1}{4}$ of the length of the underlying line-segment $||\mathbf{x}_a - \mathbf{x}_b||$.

Same as in previous studies[45], we chose the MSE loss to measure pixel difference between ground truth and predicted density maps, with loss value being calculated as follows:

$$\ell() = \frac{1}{|\mathscr{T}|} \sum_{i=1}^{|\mathscr{T}|} ||\mathscr{D}_{T_i} - \hat{\mathscr{D}}_{T_i}||_2^2 \tag{13}$$

where $\mathscr{T}$ denotes the test set and $|\mathscr{T}|$ denotes number of images in the test set. For each test image T_i, $\mathscr{D}_{T_i}$ and $\hat{\mathscr{D}}_{T_i}$ denote ground-truth and predicted density maps, respectively. $||\cdot||_2^2$ represents the Euclidean distance.

Performance metrics

We use mean absolute error (MAE) and root mean square error (RMSE) in the experiments to evaluate algorithm performance.

$$MAE = \frac{1}{|\mathscr{T}|} \sum_{i=1}^{|\mathscr{T}|} |C_{T_i} - \hat{C}_{T_i}| \tag{14}$$

$$RMSE = \sqrt{\frac{1}{|\mathscr{T}|} \sum_{i=1}^{|\mathscr{T}|} (C_{T_i} - \hat{C}_{T_i})^2} \tag{15}$$

where C_{T_i} and $\hat{C}_{T_i}$ are the ground-truth number and predicted number of manatees in image T_i, respectively, calculated using Eq. (6).

Baselines

MCNN[18], SANet[53], VGG[52], and CSRNet[45] are some of the commonly used networks for counting problem. Meanwhile, attention mechanisms were also adopted for counting task recently, such as MARUNet[54]. Those models are implemented and trained for comparisons to demonstrate the effectiveness of our proposed method. In the following, we use X-dot, X-line, and X-anisotropy to denote that X neural network is trained by using three different types of density maps, which are generated by point label with Gaussian kernel, line label with Gaussian kernel, and line label with anisotropy kernel, respectively.

- MCNN[18]: MCNN employs a multi-column CNN structure that is adaptive to detect heads of varying sizes, addressing scale variations inherent in crowd images and generating density maps for crowd estimation.

- SANet[53]: The Scale Aggregation Network employs a feature map encoder to aggregate multi-scale features from original images. Then it uses a density map estimator to fuse these features before generating high-resolution density maps. This architecture ensures robust crowd counting across varied crowd densities.
- VGG[52]: The VGG network, characterized by its depth and 3x3 convolutions, is known for its robust feature extraction capabilities and has been a foundational architecture in various visual tasks, including crowd counting.
- CSRNet[45]: CSRNet uses the simplified VGG-16 as front-end structure for feature extraction and a dilated convolution network as back-end, which can handle the scale variations in crowd counting tasks effectively.
- MARUNet[54]: MARUNet, Multi-level Attention Refined UNet, integrates a density map estimator and a crowd region recognizer, facilitating the network's focus on crowd regions and providing a strong baseline in crowd density map generation.

Experimental results

Figure 7 reports two examples for each level of the three density levels, low, medium and high with three different density maps. The first row are the original images that each image has 2, 4, 7, 11, 27 and 42 manatees separately. The second row is their Gaussian kernel density maps generated from point labels and the fourth row is their Gaussian kernel density maps generated by line labels while the sixth row is their anisotropic Gaussian kernel density maps. And third, fifth and seventh row are their corresponding estimated results from trained models. The number at the bottom right corner of each density map reports the ground-truth or estimated count of manatees in the image.

When comparing density maps from the training results with their original counterparts, as shown in Fig. 7, it shows that as the number of manatees in the image increases, AGK density maps can represent their original maps more closely compared to the other two density maps. The reason is that in high-density maps, manatees appear relatively smaller and possess similar sizes. As a result, the AGK-generated density provides a more accurate representation of their shapes with the given experimental hyper-parameters. In contrast, the density maps generated from GK-point and GK-line tend to distribute more widely rather than concentrating in specific manatees' areas.
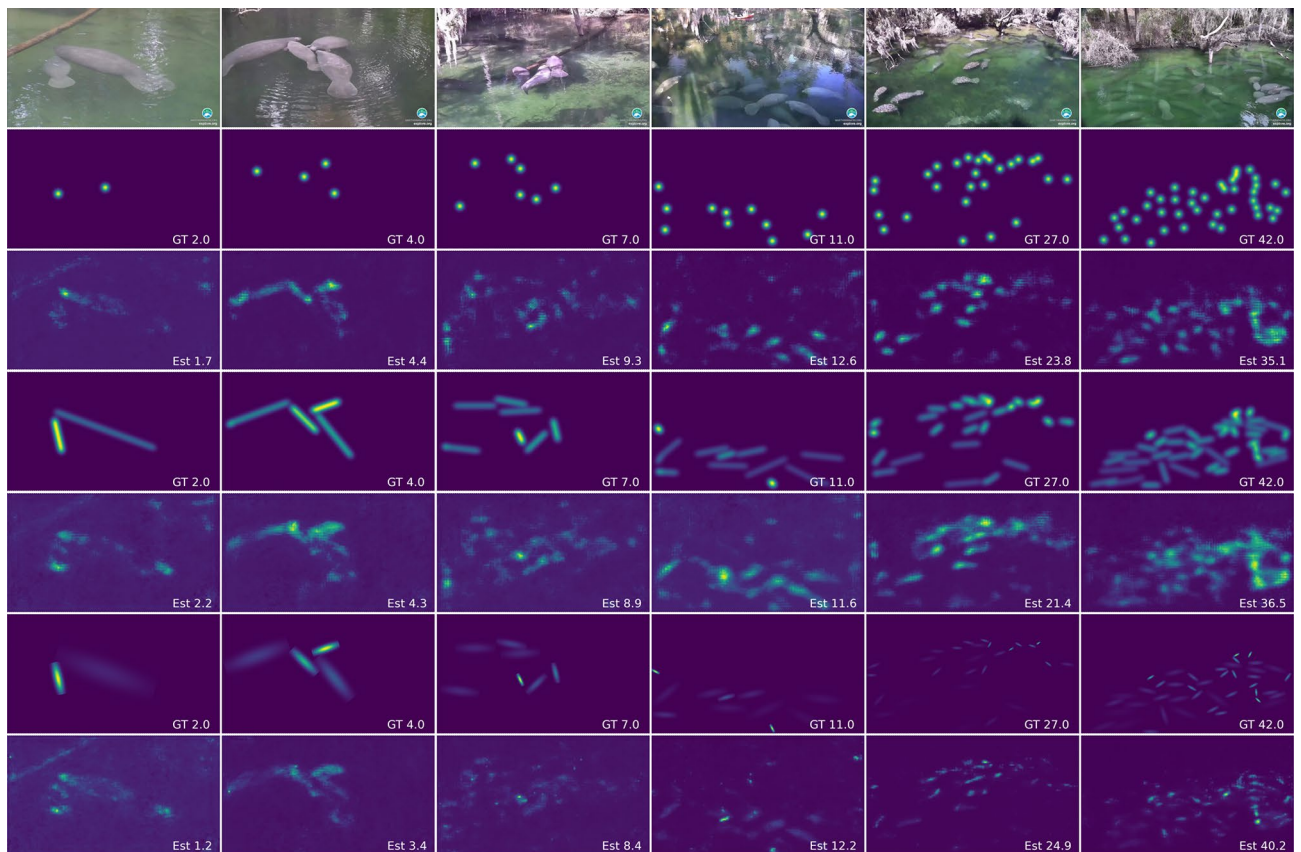

Figure 7. Examples of algorithm performance with respect to different manatee densities in the scene. The first row shows original images with increasing manatee density from left to right. The second and third rows show ground-truth density map (2nd row) and predicted density map (3rd row) using dot labels. The fourth and fifth rows show ground-truth density map (4th row) and predicted density map (5th row) using line labels and generic Gaussian kernels. The sixth and seventh rows show ground-truth density map (6th row) and predicted density map (7th row) using line labels and anisotropic Gaussian kernels.

For low and medium density maps, manatees tend to appear larger within the images. In extreme cases, a single manatee might occupy more than one-third of the entire image with only part of its body appearing in the image. Under such conditions, all three types of density maps: AGK, GK-dot, and GK-line, struggle to represent the manatees accurately, resulting in deteriorated performance. Interestingly, in these specific scenarios, GK-dot and GK-line potentially outperform AGK. This is because of their inherent capability to represent larger objects more effectively. For AGK, constrained by preset hyper-parameters, it is more adept at depicting crowded manatees, which are often of smaller sizes.

The results in Table 1 show that the anisotropy method has the lowest MAE and RMSE values across the VGG, MARUNet, and CSRNet models for the overall dataset. This shows the efficacy of our proposed line-label anisotropic Gaussian density map. Although this method does not surpass the other two when applied to the MCNN model, the performance metrics are remarkably close among all three methods. For the SANet model, the line Gaussian method yielded the most favorable results.

Among all five DNNs, SANet and CSRNet have better performance than others especially in high density maps, where MCNNs and MARUNets have extremely high MAE and RMSE values. The performance of SANet and CSRNet are very similar. In low-density scenarios, CSRNet shows slight advantages, but in medium to high-density scenarios, SANet performs better. Overall, SANet has a lower MAE while CSRNet has better stability. Interestingly, among the four compared networks, SANet is the only one that performed best under the line-label. This may be attributed to its unique scale aggregation architecture. This empirical success proves that our proposed line-label is indeed effective for crowd manatee counting task.

CSRNet-dot has the best MAE and RMSE values in low density cases while it is outperformed by CSRNet-anisotropy in high density and overall cases. Specifically, in the low density map, where manatee localizations are more distinct, CSRNet-dot registers the lowest MAE and RMSE values of 1.344 and 1.925, respectively. When the density increases, the proposed CSRNet-anisotropy shows superior MAE values in medium to high density scenarios compared to both CSRNet-dot and CSRNet-line. The superior performance of CSRNet-dot in low densities can be attributed to the clearer object locations enabled by dot annotations, given that CSRNet can effectively separate localization from counting tasks.

In terms of overall performance in CSRNet, which tested over the all data, our proposed CSRNet-anisotropy method outperforms the other two methods both in the MAE metric and in the RMSE metric, with a MAE value of 3.011 and a RMSE value of 3.962. The MAE overall performance of our proposed method is 2.99% better than the CSRNet-dot method and 21.29% better than the CSRNet-line method. And the RMSE overall performance of our proposed method is 3.63% better than the CSRNet-dot method and 23.04% better than the CSRNet-line method.

In some cases, the proposed anisotropy kernel does not consistently outperform other models, especially when combined with DNN networks like MCNN and SANet. MCNN struggles to identify the optimal method across the three different density levels. In contrast, SANet produces its best performance with the line kernel. Such discrepancies could stem from the intrinsic model structures associated with image scaling in MCNN and SANet. It becomes apparent that our method may not be universally effective for all DNN models.

Another noteworthy observation is the subpar performance of our method in low-density map levels across four out of five tested models. This might be attributed to the fact that, in low-density images, manatees often appear larger or may only partially be in the frame. Such scenarios might not be ideally represented by the density maps generated through the anisotropy kernel especially with the given experimental hyperparameters

Method	Low		Medium		High		Overall	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
MCNN-dot	3.640	3.812	**3.511**	4.924	**26.982**	**28.352**	**11.377**	12.363
MCNN-line	3.603	3.753	3.556	**4.912**	26.998	28.375	11.386	**12.346**
MCNN-anisotropy	**2.471**	**2.859**	4.170	5.562	28.080	29.408	11.573	12.610
SANet-dot	2.048	3.032	2.833	3.909	4.993	7.510	3.291	4.817
SANet-line	**1.512**	**2.645**	**2.268**	**3.418**	**4.197**	**6.882**	**2.659**	**4.315**
SANet-anisotropy	1.847	2.619	3.876	5.047	6.042	8.586	3.921	5.417
VGG-dot	3.921	4.721	2.702	3.522	15.601	17.199	7.408	8.481
VGG-line	3.723	4.540	2.678	3.535	16.217	17.748	7.539	8.608
VGG-anisotropy	**3.405**	**4.125**	**2.550**	**3.488**	**15.311**	**16.888**	**7.088**	**8.167**
MARUNet-dot	**2.312**	**2.738**	4.292	5.685	28.150	29.447	10.045	15.982
MARUNet-line	2.653	3.578	4.347	5.797	27.046	28.498	9.913	15.583
MARUNet-anisotropy	3.325	4.122	**3.830**	**5.030**	**25.063**	**26.677**	**9.567**	**14.651**
CSRNet-dot	**1.344**	**1.925**	2.981	**3.825**	4.978	6.569	3.101	4.106
CSRNet-line	2.256	3.341	3.669	4.67	5.03	6.614	3.652	4.875
CSRNet-anisotropy	1.588	2.19	**2.94**	3.84	**4.506**	**5.856**	**3.011**	**3.962**

Table 1. Manatee counting performance using different DNNs over different density levels and label types. For each method, the best result with respect to each type of labeling approach (dot, line, anisotropy) and different levels of manatee densities (low, medium, high), is bold-faced.

which used for all images. In these instances, Gaussian kernels with dot or line annotations might provide a better fitting representation.

In the subsequent section, to further substantiate the efficacy of our method, we apply the proposed method to wheat head counting, where objects have regular consistent shapes.

Additional results: wheat head counting

In the previous section, we validated our proposed method in the newly created manatee counting dataset. Given the intricate backgrounds and the variability in manatee shapes, often resulting from varying camera distances, the label can not present the manatee over different density levels as shown and analyzed in prior sections.

To further validate the ability and adaptability of our method, we have opted to test it on the "Global Wheat Head Dataset 2021"[55]. Unlike manatees, most wheat heads in these images possess consistent, elongated rectangle-like shapes. We posit that our proposed line annotation with AGK could intuitively represent such forms more effectively.

Figure 8 shows an example of original wheat head images and three corresponding density maps that are generated by a dot-based Gaussian Kernel(GK), a line-based GK and a line-based anisotropic GK. It is obvious that line-based AGK density map has a better representation of the wheat heads in terms of position, shape, and direction.

As illustrated in Table 2, Yolo-v5(GWC-2021)[56] has the poorest performance in terms of RMSE. All three models based on CSRNet outperform Yolo-v5, with CSRNet-anisotropy has outstanding MAE of 15.7 and RMSE of 20.2. Yolo-v5 is detection based algorithm that while it has good bounding box accuracy, it misses numerous wheat heads, especially those that are smaller in size. The average size of those missed wheat heads is about 35% smaller than the average size of all wheat heads.

For our density estimation-based method, instead of seeking the precise location and boundary of each wheat head directly, it determines the total number of wheat heads by estimating the density of wheat heads within a particular region. This method is particularly good at detecting smaller and densely distributed wheat heads since it emphasizes the overall quantity estimation rather than pinpointing individual targets.

When the MAE and RMSE values are close, it often indicates there are not significant outliers. Such consistency can suggest that the model's predictions are relatively accurate, with minimal discrepancies. As shown in Table 2, the proximity of the MAE and RMSE values reinforces this notion of accuracy in the CSRNet's predictions, especially CSRNet-anisotropy which proves the effectiveness of our method.

Throughout the training process, we noticed that the model effectively capture the macro structure of the wheat. For point label, the model had to work hard to fit the density map precisely to a circular region. To address this challenge, we introduced AGK to generate density map by using line labels, which differ significantly from the traditional circular Gaussian region. This label utilizes an elliptical anisotropic Gaussian distribution to present the wheat's location and shape. In this distribution, the core line represents the exact location of the wheat head, while the surrounding decay area outlines the potential uncertainty of the position. Compared to the conventional circular Gaussian region, this method endows the model with enhanced adaptability in wheat head counting.
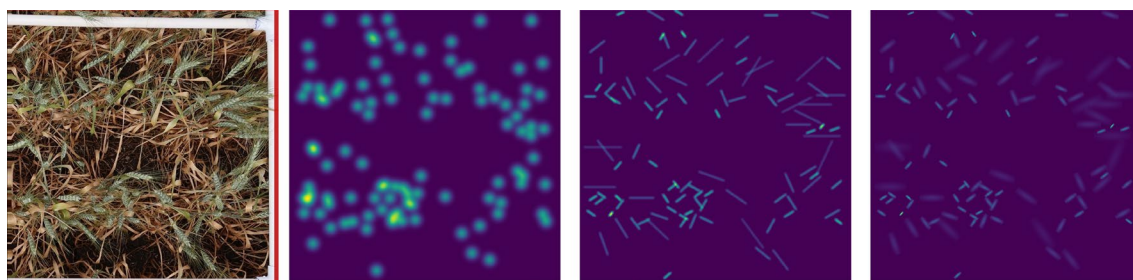

Figure 8. The original wheat image and its three types of corresponding density maps (left to right): dot based GK, line based GK, and line based AGK.

Method	MAE	RMSE
Yolo-v5(GWC-2021)[56]	NA	67.0
CSRNet-dot	30.1	37.6
CSRNet-line	20.3	28.1
CSRNet-anisotropy	15.7	20.2

Table 2. Wheat head counting performance in CSRNet over different label types.

Conclusion

In this paper, we proposed a deep neural network (DNN) based crowd counting approach to count manatee aggregations. This method capitalizes on low-quality images to count manatees in a designated region. Although crowd counting has been used in many other applications (e.g. counting cars or audiences), we argued that manatee counting has unique challenges, including surface reflection, occlusions, camouflage. To reduce labelling costs, we employed line-label based annotation, with a single straight line being used to mark each manatee. To take unique shapes of manatees into consideration, we proposed to use Anisotropic Gaussian Kernel (AGK) transform input images into manatee customized density maps, and then train deep neural networks to learn to count manatee numbers automatically using a predicted density map. Experiments and comparisons, using low resolution real-world manatee images, show that AGK based counting outperforms other baselines, including traditional Gaussian kernel based approach. The proposed approach works particularly well when the image has a high density of manatees in complicated background.

Our findings demonstrate a promising trajectory for broader applications. By transitioning from dot to line labeling, we not only enhanced the accuracy in counting manatees but also successfully improved wheat head counting. The proposed methodology holds potential for various applications, especially for entities with convex-shaped objects, including diverse animals such as livestock like sheep and cattle, and crops such as wheat head, corn, eggplant, etc.

In this study, we are primarily focused on images captured from the water surface. Counting manatees in complex underwater backgrounds remains an open problem[57]. Future study may take manatees' movement into consideration to improve counting accuracy, as static objects like branches and rocks remain relatively unchanged over short duration.

Data availability

Source code and datasets generated and/or analysed in this study are available in the following GitHub repository: https://github.com/yeyimilk/deep-learning-for-manatee-counting.

References

1. Zhu, X. & Davidson, I. *Knowledge Discovery and Data Mining: Challenges and Realities* (Information Science Reference, 2007).
2. Corcoran, E., Denman, S., Hanger, J., Wilson, B. & Hamilton, G. Automated detection of koalas using low-level aerial surveillance and machine learning. *Sci. Rep.* **9**, 1–9 (2019).
3. Banks, N. C., Paini, D. R., Bayliss, K. L. & Hodda, M. The role of global trade and transport network topology in the human-mediated dispersal of alien species. *Ecol. Lett.* **18**, 188–199 (2015).
4. Ceballos, G. *et al.* Accelerated modern human-induced species losses: Entering the sixth mass extinction. *Sci. Adv.* **1**, e1400253 (2015).
5. Quintana-Rizzo, E. & Reynolds, J. III. *Regional management plan for the West Indian manatee (Trichechus manatus)* (Kingston, UNEP Caribbean Environment Programme, 2010).
6. Henaut, Y., Charles, A. & Delfour, F. Cognition of the manatee: Past research and future developments. *Anim. Cogn.* 1–10 (2022).
7. van Roosmalen, M. G. Hotspot of new megafauna found in the central amazon (brazil): The lower rio aripuanã basin. *Biodivers J6*, 219–244 (2015).
8. Vianna, J. A. *et al.* Phylogeography, phylogeny and hybridization in trichechid sirenians: Implications for manatee conservation. *Mol. Ecol.* **15**, 433–447 (2006).
9. Deutsch, C., Self-Sullivan, C., Mignucci-Giannoni, A. *et al.* Trichechus manatus. *The IUCN Red List of Threatened Species* **2008** (2008).
10. Nourisson, C. *et al.* Evidence of two genetic clusters of manatees with low genetic diversity in Mexico and implications for their conservation. *Genetica* **139**, 833–842 (2011).
11. Castelblanco-Martínez, D. N. *et al.* First documentation of long-distance travel by a Florida manatee to the Mexican Caribbean. *Ethol. Ecol. Evol.* 1–12 (2021).
12. Corona-Figueroa, M. F. *et al.* Searching for manatees in the dark waters of a transboundary river between Mexico and Belize: A predictive distribution model. *Aquat. Ecol.* **55**, 59–74 (2021).
13. Rodas-Trejo, J., Romero-Berny, E. I. & Estrada, A. Distribution and conservation of the west Indian manatee (Trichechus manatus manatus) in the catazajá wetlands of Northeast Chiapas, México. *Trop. Conserv. Sci.* **1**, 321–333 (2008).
14. Keith Diagne, L. Trichechus senegalensis. the iucn red list of threatened species 2015: e. t22104a97168578 (2015).
15. Edwards, H. H., Hostetler, J. A., Stith, B. M. & Martin, J. Monitoring abundance of aggregated animals (Florida manatees) using an unmanned aerial system (USA). *Sci. Rep.* **11**, 12920 (2021).
16. Bauduin, S. *et al.* An index of risk of co-occurrence between marine mammals and watercraft: Example of the Florida manatee. *Biol. Cons.* **159**, 127–136 (2013).
17. Arteta, C., Lempitsky, V. & Zisserman, A. Counting in the wild. In *European conference on computer vision*, 483–498 (Springer, 2016).
18. Zhang, Y., Zhou, D., Chen, S., Gao, S. & Ma, Y. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 589–597 (2016).
19. Zhang, C., Li, H., Wang, X. & Yang, X. Cross-scene crowd counting via deep convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 833–841 (2015).
20. Pang, Y., Ni, Z. & Zhong, X. Federated Learning for Crowd Counting in Smart Surveillance Systems. In *IEEE Internet of Things Journal*. https://doi.org/10.1109/JIOT.2023.3305933 (2023).
21. Enzweiler, M. & Gavrila, D. M. Monocular pedestrian detection: Survey and experiments. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**, 2179–2195 (2008).
22. Leibe, B., Seemann, E. & Schiele, B. Pedestrian detection in crowded scenes. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, 878–885 (IEEE) (2005).
23. Wu, B. & Nevatia, R. Detection of multiple, partially occluded humans in a single image by Bayesian combination of edgelet part detectors. In *10th IEEE Intl. Conf. on Computer Vision (ICCV'05) Volume 1*, vol. 1, 90–97 (IEEE, 2005).
24. Gao, G., Gao, J., Liu, Q., Wang, Q. & Wang, Y. Cnn-based density estimation and crowd counting: A survey. arXiv preprint arXiv:2003.12783 (2020).

25. Topkaya, I. S., Erdogan, H. & Porikli, F. Counting people by clustering person detector outputs. In *2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 313–318 (IEEE, 2014).
26. Dollar, P., Wojek, C., Schiele, B. & Perona, P. Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**, 743–761 (2011).
27. Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788 (2016).
28. Ren, S., He, K., Girshick, R. & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **28** (2015).
29. Farjon, G., Huijun, L. & Edan, Y. Deep-learning-based counting methods, datasets, and applications in agriculture: A review. *Precis. Agric.* 1–29 (2023).
30. Rahnemoonfar, M. & Sheppard, C. Deep count: Fruit counting based on deep simulated learning. *Sensors* **17**, 905 (2017).
31. Bai, X. *et al.* Rpnet: Rice plant counting after tillering stage based on plant attention and multiple supervision network. *Crop J.* (2023).
32. Lin, Y. *et al.* Transfer Learning on the Feature Extractions of Sky Images for Solar Power Production. In *2019 IEEE Power & Energy Society General Meeting (PESGM)*, 1–5. https://doi.org/10.1109/PESGM40551.2019.8973423 (2019).
33. de Lima Weber, F. *et al.* Counting cattle in uav images using convolutional neural network. *Remote Sens. Appl. Soc. Environ.* **29**, 100900 (2023).
34. Barbedo, J. G. A., Koenigkan, L. V., Santos, T. T. & Santos, P. M. A study on the detection of cattle in uav images using deep learning. *Sensors* **19**, 5436 (2019).
35. Rahnemoonfar, M., Dobbs, D., Yari, M. & Starek, M. J. Discountnet: Discriminating and counting network for real-time counting and localization of sparse objects in high-resolution uav imagery. *Remote Sens.* **11**, 1128 (2019).
36. Chan, A. B. & Vasconcelos, N. Bayesian poisson regression for crowd counting. In *2009 IEEE 12th international conference on computer vision*, 545–551 (IEEE, 2009).
37. Pham, V.-Q., Kozakaya, T., Yamaguchi, O. & Okada, R. Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. In *Proc. of the IEEE Intl. Conf. on Computer Vision*, 3253–3261 (2015).
38. Borstel, M. v. *et al.* Gaussian process density counting from weak supervision. In *European Conference on Computer Vision*, 365–380 (Springer, 2016).
39. Yang, Y. *et al.* Weakly-supervised crowd counting learns from sorting rather than locations. In *European Conference on Computer Vision*, 1–17 (Springer, 2020).
40. Lei, Y., Liu, Y., Zhang, P. & Liu, L. Towards using count-level weak supervision for crowd counting. *Pattern Recogn.* **109**, 107616 (2021).
41. Sam, D. B., Sajjan, N. N., Maurya, H. & Babu, R. V. Almost unsupervised learning for dense crowd counting. *In Proceedings of the AAAI Conference on Artificial Intelligence* **33**, 8868–8875 (2019).
42. Wang, F. *et al.* Joint cnn and transformer network via weakly supervised learning for efficient crowd counting. arXiv preprint arXiv:2203.06388 (2022).
43. Liang, D., Chen, X., Xu, W., Zhou, Y. & Bai, X. Transcrowd: Weakly-supervised crowd counting with transformers. *Sci. China Inf. Sci.* **65**, 1–14 (2022).
44. Wang, M., Zhou, J., Cai, H. & Gong, M. Crowdmlp: Weakly-supervised crowd counting via multi-granularity mlp. arXiv preprint arXiv:2203.08219 (2022).
45. Li, Y., Zhang, X. & Chen, D. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1091–1100 (2018).
46. Liu, W., Salzmann, M. & Fua, P. Context-aware crowd counting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 5099–5108 (2019).
47. Shi, M., Yang, Z., Xu, C. & Chen, Q. Revisiting perspective information for efficient crowd counting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7279–7288 (2019).
48. Yan, Z. *et al.* Perspective-guided convolution networks for crowd counting. In *Proceedings of the IEEE/CVF international conference on computer vision*, 952–961 (2019).
49. Lian, D., Li, J., Zheng, J., Luo, W. & Gao, S. Density map regression guided detection network for rgb-d crowd counting and localization. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 1821–1830 (2019).
50. Shamsheyeva, A. & Sowmya, A. The anisotropic gaussian kernel for svm classification of hrct images of the lung. In *Proc. of the Intelligent Sensors, Sensor Networks and Info. Proc. Conf.*, 439–444 (2004).
51. Kim, J. & Scott, C. D. Robust kernel density estimation. *J. Mach. Learn. Res.* **13**, 2529–2565 (2012).
52. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014).
53. Cao, X., Wang, Z., Zhao, Y. & Su, F. Scale aggregation network for accurate and efficient crowd counting. In *Proceedings of the European conference on computer vision (ECCV)*, 734–750 (2018).
54. Rong, L. & Li, C. Coarse-and fine-grained attention network with background-aware loss for crowd density map estimation. In *Proc. of the IEEE/CVF Conf. on Applications of Computer Vision*, 3675–3684 (2021).
55. David, E. *et al.* Global wheat head dataset 2021: more diversity to improve the benchmarking of wheat head localization methods. arXiv preprint arXiv:2105.07660 (2021).
56. David, E. *et al.* Global wheat head detection challenges: Winning models and application for head counting. *Plant Phenom.* (2023).
57. Zhou, W., Zheng, F., Yin, G., Pang, Y. & Yi, J. Yolotrashcan: A deep learning marine debris detection network. *IEEE Trans. Instrum. Meas.* **72**, 1–12. https://doi.org/10.1109/TIM.2022.3225044 (2023).

Acknowledgements

Author contributions

Drafting of the manuscript: Z.W., Y.P., C.U. Design and modeling: Z.W., Y.P., C.U., X.Z. Data collection and analysis: Z.W., Y.P., C.U. Obtained funding: X.Z. Supervision: X.Z.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to X.Z.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.