Distance Estimation and Animal Tracking for Wildlife Camera Trapping

Peter Johanns, Timm Haucke, Volker Steinhage

*University of Bonn, Institute of Computer Science IV, Friedrich-Hirzebruch-Allee 8, Bonn 53115, Germany

Abstract

The ongoing biodiversity crysis calls for accurate estimation of animal density and abundance to identify, for example, sources of biodiversity decline and effectiveness of conservation interventions. Camera traps together with abundance estimation methods are often employed for this purpose. The necessary distances between camera and observed animal are traditionally derived in a laborious, fully manual or semi-automatic process. Both approaches require reference image material, which is both difficult to acquire and not available for existing datasets. In this study, we propose a fully automatic approach to estimate camera-to-animal distances, based on monocular depth estimation (MDE), and without the need of reference image material. We leverage state-of-the-art relative MDE and a novel alignment procedure to estimate metric distances. We evaluate the approach on a zoo scenario dataset unseen during training. We achieve a mean absolute distance estimation error of only 0.9864 meters at a precision of 90.3% and recall of 63.8%, while completely eliminating the previously required manual effort for biodiversity researchers. The code will be made available.

Keywords: Animal density, animal abundance, camera trapping, unmarked animal populations, automated distance estimation, animal tracking

*Email addresses: johanns@uni-bonn.de (Peter Johanns), haucke@cs.uni-bonn.de (Timm Haucke), steinhage@cs.uni-bonn.de (Volker Steinhage)
1. Introduction

The biodiversity crisis necessitates an accurate monitoring of animal density and abundance. Such estimates can then be used to identify causes of biodiversity loss and quantify the effects of conservation efforts. This is often achieved by employing camera traps, which capture image or video upon detection of an animal by a passive infrared sensor. Capture-recapture models can be used to estimate animal abundance by (re-)identifying individual animals over multiple images (O’Connell et al., 2011), which is however difficult for species without individual markings.

1.1. Related Work

**Abundance Estimation:** In response, several abundance estimation methods for unmarked animal populations have been developed, which do not require the identification of individuals: the random encounter model (REM) (Rowcliffe et al., 2008), the random encounter and staying time model (REST) (Nakashima et al., 2018), the time-to-event model (TTE), space-to-event model (STE), instantaneous estimator (IS) (Moeller et al., 2018) and camera trap distance sampling (CTDS) (Howe et al., 2017). While these methods do not require the reidentification of individual animals, they do require an estimation of the effective area surveyed by the camera trap. The effective surveyed area is dependent on the Field of View (FOV) of the camera trap and its effective detection distance. The effective detection distance is the distance below which as many individuals are missed as are seen beyond (Hofmeester et al., 2017). Estimating the effective detection distance generally requires estimating the distance between the camera trap and the detected animals. So far, three different approaches are available to derive such camera-animal distances. However, they rely either on the manual and laborious evaluation of reference images (Howe et al., 2017), even more time-consuming on-site distance measurements (Rowcliffe et al., 2011) or semi-automatic calibration of relative depth images for the specific sequence (Haucke et al., 2022).

**Depth Estimation:** Depth estimation is the process where for every pixel in a color input image the distance of the corresponding object to the camera is estimated and saved in a congruent depth image (Fig. 1). Depth images are usually provided by specific sensors like stereo cameras or LiDAR (Light Detection And Ranging) scanners. These sensors are often expensive or not available in most scenarios, therefore Monocular Depth Estimation (MDE) on color images or videos captured by a single camera is the only viable alternative. MDE has a wide variety of use cases and can be used for an almost unlimited amount of tasks, for example, augmented reality applications (Woo et al., 2011), real-time 3D human pose and movement estimation (Moon et al., 2017), navigation of autonomous vehicles (Geiger et al., 2013) or robots, 3D photography (Kopf et al., 2020), 3D scene reconstruction (Fig. 10) and many more. In this work, we focus on the application of MDE to abundance estimation using camera trap images.
1.2. Problem Statement & Contributions

Objective

- Our objective is the robust distance estimation and tracking of animals observed by camera trapping. This objective comprises three aspects: (1) Locating observed animals, (2) estimating the distances of observed animals to the camera trap in meters and (3) associating the spatial positions of observed animals across the camera trap video clips resulting in coherent spatial tracks of the observed animals.

Problem Statement

- To our best knowledge, there is no fully automated approach to the given objective. Instead, current approaches demand for generating reference data including time consuming manual preparation, interaction and post-processing of these data material.

- End-to-end deep learning models have demonstrated state-of-the-art performance in Machine Learning, showing several advantages with respect to development, optimization, maintenance, and applicability to new tasks (Bojarski et al., 2016; Wang et al., 2019; Mueller and Chi, 2022). However, training of an end-to-end deep learning model for simultaneously performing identification and distance estimation of observed animals is challenging due to lack of datasets that combine (1) animal tracks as well as (2) depth imagery of the observed animals.

- State-of-the-art depth estimation models only yield relative distance estimations. The required conversion to absolute distance values in meters is currently performed only manually or semi-automatically (cf. section 1.1).

Contributions

- To our best knowledge, we propose the first fully automated processing pipeline for (1) the visual detection of animals observed by camera trapping, (2) the derivation of their metric distances to the camera trap and (3) the tracking of their three-dimensional movements relative to the camera. Thereby, our approach is not demanding for prerecorded reference material nor depends on manual intervention by the scientists.

- Our proposed pipeline is based on a two-branch architecture (Fig. 1) that allows end-to-end learning of both sub-tasks, i.e. (1) animal identification and tracking and (2) distance estimation. Additionally, this architecture shows a modularity that
  - allows adaption and replacement of modules to improve performance,
  - takes advantage of dataset abundance, i.e. allows for the exploitation of a rich diversity of datasets for training and testing,
– yields depth images of the complete observed scenarios and not just of observed animals yielding additional opportunities like background modelling, background subtraction etc.

• Of particular importance is the novel alignment procedure that automatically estimates the parameters to convert relative depth imagery into metric depth imagery.

2. Materials and Methods

In this section, we explain each module of the proposed pipeline and its functionality as well as the selection criteria that were used. Furthermore we give an overview of the selected training and testing datasets and their properties.

2.1. Data Material

We want to train a model that is capable of achieving accurate distance estimation results on camera trap data from datasets not seen during training (Zero-Shot-Evaluation). Therefore, a diverse collection of outdoor depth training data is necessary to promote the generalizability of the model. The datasets were selected under the following criteria:

**Natural scenes:** Datasets are often human-centric and therefore contain highly regular geometry such as streets and buildings. In contrast, camera traps are often installed in natural environments such as forests, which contain no such geometry. The training data for the alignment model should therefore include data from as many natural environments as possible.

**Absolute depth information:** The task of the alignment model is to convert relative into absolute depth information. To train the model on this task in a supervised setting, the training depth data must have an absolute scale.

**Known field of view:** The alignment model needs to create an internal 3D point-cloud representation. To do this, the opening angle, or field of view, of the camera setups used for training must be known.

We settled on the following collection. Image samples can be found in Fig. A.11 and an overview of their properties in Table 1:

**UASOL** (Bauer et al., 2019) is a stereo dataset recorded from the point of view of a pedestrian around the campus of the University of Alicante (Spain). Due to the size, we only selected five of 33 available scenes: EPS4, Garden, Nursery, Optics, and Philosophy 1, with the largest amount of visible vegetation.

**TartanAir** (Wang et al., 2020) is a photo-realistic synthetic dataset captured from the point of view of a flying drone and rendered with Unreal Engine. The authors present drone flight paths of varying complexity; we decided to use the simplest flight patterns and selected five of the 30 given scenes. These were recorded in the following outdoor environments: Gascola, Neighborhood, Seasons Forest and Seasons Forest Winter.
DIML (Cho et al., 2021) is a RGBD-dataset consisting of more than 200 different indoor and outdoor scenes recorded with a Microsoft Kinect and a ZED stereo camera. Due to availability and memory constraints we decided to only use the Scenes Field 1 and Field 2, as these depicted scenes are comparable to camera trap videos. DIML serves as the validation dataset during training.

LVPD (Niu et al., 2020) is a forest environment dataset collected in woodland areas in Southampton Common (Hampshire, UK). The camera was mounted 15 cm above the ground on a broom-like contraption to simulate the point of view of a robotic ground rover. The images provided by this dataset were most similar to real world camera trap videos of a camera mounted to a tree in a dense forest biosphere.

Lindenthal is, to our knowledge, the only outdoor dataset that provides depth as well as tracking annotations for animals (Haucke and Steinhage, 2021). We therefore use it as the dataset for zero-shot evaluation. It was recorded by an Intel RealSense D435i stereo camera which was mounted above an animal enclosure at the Lindenthal Zoo (Cologne, Germany) and recorded scenes of passing geese, goats, donkeys and deer in gray scale video at 15 frames per second. There is a total amount of 14 scenes, which we enumerated from S00 to S13 (Table A.5). Every tenth frame of each sequence was annotated with tracked bounding boxes and segmentation masks for visible animals. A detailed example for the annotation can be found in Figure 9.

All datasets except TartanAir include depth images with invalid pixel values as a result of “infinite” distances (e.g. sky) or the stereo camera setup. The invalid pixels are masked out during training and evaluation and are depicted white in Fig. A.11. The datasets were recorded with devices with diverse depth sensing ranges. To decrease the negative impact of decreasing depth accuracy for long distances on the training data and to lower the possibility of instability due to large values, pixels with an assigned distance of more than 65 meters were masked out as well.

| Dataset   | Frames | Scenes | Resolution        | Acquisition | Video | Focal Length [px] | HFOV [degree] | Max Depth [m] |
|-----------|--------|--------|-------------------|-------------|-------|-------------------|---------------|---------------|
| UASOL     | 25.7 K | 5      | 2208 x 1242       | Stereo ZED | yes   | 1399.74           | 76.5          | 20            |
| TartanAir | 24.5 K | 5      | 640 x 480         | Rendered    | yes   | 320               | 90            | 65            |
| DIML      | 9.7 K  | 14     | 848 x 480         | Stereo RealSense D435i | 462.14 | 74                | 10            |
| LVPD      | 5.8 K  | 14     | 640 x 480         | Stereo RealSense D435i | yes   | 424.74            | 90            | 65            |

Table 1: Datasets Characteristics Overview: Dataset Name, Number of Images, Number of Scenes, Resolution of RGB and Depth-Images, Acquisition of Images, Scenes are Videos, Focal Length [px], Horizontal Field of View [degree], Maximum Depth [m]

### 2.2. Developing the Pipeline

The pipeline (Fig. 1) accepts a RGB video sequence split into frames as input. In the left branch, the relative depth is computed for every frame using the DPT-Monodepth model (Dense Prediction Transformer, Section 2.2.1 (Ranftl et al., 2021)). The relative depth image without a metric scale is then aligned
Figure 1: Pipeline Architecture: The input are a RGB video clips, the output consists of the 3D tracks of the observed animals. Newly developed modules are are highlighted in green colour. The circles depict the corresponding subsections in the section of methods (red) and in the evaluation section (blue), respectively.
by the PVCNN-module (Point-Voxel Convolutional Neural Network, Section 2.2.2 (Liu et al., 2019)) to contain values in meters. In the right branch, we extract the animal positions in each frame by using the MegaDetector-framework (Section 2.2.3 (Beery et al., 2019)) that outputs the bounding boxes of every detection (red rectangles). For every bounding box, we now want to extract as many pixels as possible that belong to the animal itself and not the background. To achieve this, we employ the DINO model (self-DIstillation with NO labels, Section 2.2.4 (Caron et al., 2021)) in a multi-instance approach to extract a binary foreground mask, the pixels that depict the animal and not the background, per identified animal. We now have to marry the results of the two branches to obtain detections with their depth information. We achieve that by applying the foreground mask of every detection to the corresponding depth image of the upper branch and by determining the median of the selected pixel depth values as the depth of the detection. In the last step a modified version of the SORT tracking algorithm (Simple Online Real-Time Tracker, Section 2.2.5 (Bewley et al., 2016)) is employed to calculate cohesive 3D tracks from the single detections. The resulting tracks can be visualized by unprojecting the 2D RGB images back to 3D (Fig. 10).

2.2.1. Relative Depth: DPT

For the relative depth estimation, we use the DPT (Dense Prediction Transformer) model developed by (Ranftl et al., 2021). Combined with a large amount of diverse depth datasets, the authors achieve a new state-of-the-art performance during the evaluation on unseen datasets and thus create a robust model for a wide variety of scenes. However, the model only estimates the relative disparity and not the absolute metric depth in meters to avoid instability due to the wide range of possible depth scales in the training data.

Adapting DPT: In the pipeline each input frame is processed by DPT to a disparity image \( \hat{d} \) and then converted to an approximated depth image \( d \) (Eq. 1). We determined the necessary conversion parameters scale \( m = 0.5 \) and shift \( c = 0.02 \) in the disparity space by aligning each DPT output of every image in the training dataset to its Groundtruth via RANSAC (Random Sample Consensus, Section 3.1) and averaging across the resulting scales and shifts.

\[
\overline{d} = \frac{1}{d \cdot m + c}
\]  

(1)

2.2.2. Robust Distance Estimation: PVCNN

We want to calculate a metric depth estimation for an input color image. To this end, we have to align the approximated depth output of DPT \( \overline{d} \) with a scale \( m \) and a shift \( c \) to a metric depth image \( d_m \), such that

\[
d_m = m \cdot \overline{d} + c
\]

(2)

The scale \( m \) determines the visible range of depth values, while the shift \( c \) determines the distance of the closest object to the camera and the lowest value of the depth range. To calculate both parameters, we adapt and modify the
approach proposed by (Yin et al., 2021). The authors’ goal is to recover the 3D scene shape from a single image. To accomplish this, they estimate a relative depth image, convert it to a point cloud representation, and then utilize a Point-Voxel-CNN (PVCNN) (Liu et al., 2019) to estimate the focal length and the shift needed to create an unwarped 3D reconstruction of the scene.

Adapting PVCNN: We extend the PVCNN architecture to estimate both scale $m$ and shift $c$, as well as by introducing extensive data augmentation and a novel training regime.

Data Augmentation. We want to create a model that performs well on images not seen during training and that does not simply remember the training images (“overfitting”). To this end, we apply random modifications to the training data during runtime, so that the model receives slightly different inputs each epoch, which overall increases the amount of provided training samples (“augmentation”). We apply the following augmentation steps to the approximated depth images $d$ to improve generalization to unseen resolutions, scenes and focal lengths:

- Random horizontal flip with a probability of 0.5
- Random crop to 16:9 or 4:3 aspect ratio
- Select random factor $s \in [0.75, 1]$; Centered Crop of image of the original size multiplied by $s$; Resize to original resolution; Multiply focal length by $\frac{1}{s}$ and the depth ground truth by $s$

Training Regime. Furthermore, for our PVCNN we develop the following training regime (Fig. 2): First, we unproject the relative depth image back to 3D similar to (Yin et al., 2021). We assume a pinhole camera for the point cloud reconstruction and convert 2D image coordinates to 3D by:
where \((u_0, v_0)\) is the optical center of the camera, \(f\) is the focal length, and \(\bar{d}\) is the approximated depth. Contrary to (Yin et al., 2021), we presume that the focal length of the camera is known and do not seek to calculate it. After that, the point cloud is converted to a voxelized version of a certain resolution and given to the PVCNN as input. From there on, the PVCNN estimates the needed scale and shift for the input image. To calculate the training loss, we align the initial approximated depth input with the scale and shift and apply our loss function (Eq. 4) to the output and the ground truth. Simply calculating a common pixel-wise loss would have the disadvantages that the model would minimize the loss for all pixels equally and would be susceptible to either outliers in the depth estimation of DPT (especially for high distances) or to errors in the ground truth. Instead, we propose a weighted loss function that shifts the learning objective to pixels closer to the camera. Let \(d_m\) be the aligned metric depth image, \(g\) the depth ground truth image, \(n_{valid}\) the number of valid pixels in the ground truth, \(\exp\) the exponential function and \(\alpha\) the weight factor, then we define the weighted loss \(L_w\) as:

\[
L_w(d_m, g) = \frac{(d_m - g)^2 \cdot \exp(-\alpha \cdot g)}{n_{valid}}
\]  

(4)

The factor \(\alpha\) controls how much closer pixels with lower depth values influence the overall loss. As a result, we set \(\alpha\) to 0.04 during training, as it achieved the best results on the validation data. For each image, we divide the calculated loss by \(n_{valid}\), as invalid pixels are masked out during training and images from different datasets display varying numbers of valid pixels.

Due to the fact that all datasets contain a different number of images, we decide to randomly sample 40K images per training epoch from all datasets, in a way that each batch contains an equal number of images from each training dataset. For an efficient batch-wise loss calculation, all ground truth images as well as the model output are resized to a resolution of 640x480 using nearest-neighbour interpolation. We train our model for seven epochs on batches of 50 images, with a voxel size of 0.01, we sample 25K points per input image to avoid overfitting on the number of points, employ a learning rate of 0.0001 with a decay factor of 0.1 applied every fourth epoch and we train with a dropout probability of 0.3 for the classifier layer.

Directly training on the scales and shifts calculated using RANSAC yielded inferior results during training. Furthermore, we decided against an approach that estimates the two parameters directly from images or image features. As (Yin et al., 2021) observed, “the domain gap is significantly less of an issue for point clouds than for images” for this kind of task, which requires a accurate 3D reconstruction of the scene.
2.2.3. Animal Detection: MegaDetector

The MegaDetector is an animal detection model for camera trap footage developed by (Beery et al., 2019) and was trained on several hundred thousand animal detections from camera trap videos recorded in diverse biospheres and of a large variety of animals. This detection model was selected because of its robustness: It is able to localize animals and species not seen during training and it reliably detects animals in unseen eco-systems and weather conditions as well.

2.2.4. Foreground Segmentation: Multi-Instance DINO

![Image](image.png)

**Figure 3:** Original Detection (a); Wider Region around Detection with more Context (b), DINO Attention Map Output (c), Original Detection with Foreground Segmentation (d)

DINO (Caron et al., 2021), self-DIstillation with NO labels, describes a method of pretraining a visual transformer (ViT) without supervision to output self learned class features and attention maps (Fig. 3 (c)). DINO was selected as the foreground segmentation model because it requires no additional training, correctly handles partial occlusion by vegetation and provides a precise segmentation result. As only the majority of marked pixels must belong to the animal, a complete mask is not necessary. The attention maps are used to mask out a critical amount of pixels belonging to a detected animal inside the bounding box (Fig. 3). As DINO was trained on the ImageNet dataset (Deng et al., 2009), which mostly contains images, where only one relevant object is in the center of the frame, simply applying DINO to a complete image has two disadvantages: The model does not recognize every animal and only produces attention maps of a low resolution.

**Adapting DINO:** Instead, we combine DINO with MegaDetector detections and propose Multi-Instance DINO. Specifically, we take the following steps (Fig.3):

1. Input: The bounding box of a detection (a)
2. Increase the crop around the detection by doubling the bounding box measurements (b)
3. Generate the attention map of the widened crop using DINO (c)
4. Create a foreground segmentation mask using the minimum intensity threshold of 10.2% (d)

The marked pixels are then used to determine the distance of the animal by taking the median of the corresponding depth pixel values in the aligned depth.
image. By using widened and centered crops of the objects of interest, we mimic the format of the ImageNet dataset and as a consequence create optimal input images for DINO to operate on.

2.2.5. Tracking: SORT 2.5D

Calculating tracks from the detections with corresponding depth is the last processing step of the proposed pipeline. To this end, we use the Simple Online Realtime Tracking (SORT) algorithm (Algorithm 1) developed by Bewley et al. (2016) that takes 2D bounding box detections as input. SORT connects the detections over all frames to cohesive tracks based on a Kalman-Filter framework and the association metric of Intersection over Union (IoU) (Jaccard, 1912). We decided for the use of this simple tracking approach and against an end-to-end deep learning model, as SORT required no additional training and was easy to implement. Moreover, the possible performance gain of a deep-learning-based tracker is questionable, due to the fact that in camera trap videos individuals of the same species are hard to tell apart. When the track association based on visual similarity is not possible, a method purely dependant on the location and speed of a detection like SORT is the best choice.

Adapting SORT: We modify the algorithm to include the depth in the Kalman-Filter predictions and replace IoU with our custom association metric SimScore [5] that combines the traditional IoU with a distance similarity metric DISTZ. \( \alpha \) controls the weight of each metric. DISTZ depends on the hyperparameter DISTmax. If the depth distance between the tracker prediction \( z_T \) and the detection \( z_{DET} \) is bigger than DISTmax, DISTZ is clipped to zero, otherwise the difference is substracted from DISTmax and then normalized. Going forward, we will refer to the modified SORT version as SORT 2.5D.

\[
SimScore = \alpha \cdot IoU + (1 - \alpha) \cdot DISTZ, \quad \alpha \in [0, 1] \tag{5}
\]

\[
DISTZ = \left( \frac{DIST_{max} - |z_T - z_{DET}|}{DIST_{max}} \right) .clip(0, 1) \tag{6}
\]

To tune the hyperparameters for the approach we conducted a random search of 40K iterations on the sequence S09 using the MOTA score and determined the values seen in Table 2. The MOTA metric was selected to be optimized, as it is the closest representation of the perceived tracking quality of a Multiple Object Tracker (Yu et al., 2016).

| Parameter                  | 2.5D SORT |
|----------------------------|-----------|
| CONFmin                    | 0.916     |
| AGEmax                     | 111       |
| HITSmint                   | 1         |
| IOU \(_{min} \) / SIMSCOREmin | 0.01     |
| \( \alpha \)               | 0.427     |
| DISTmax                    | 4.096     |

Table 2: Hyperparameter for 2.5D SORT after random search on S09
3. Implementation, Evaluation and Discussion

In this section, we conduct the evaluation of our proposed pipeline. First, we examine the performance of the distance estimation module through a zero-shot evaluation on the Lindenthal-dataset. We measure the model’s precision on the complete depth images and on the animal detections. Additionally, we point out failure cases of the DINO Segmentation by showing qualitative examples. To determine the tracking quality of our SORT 2.5D modification, we used the so-called MOT (Multiple Object Tracker) metrics with the matching criterion of a maximum allowed distance to the ground truth of \( \sqrt{5} \approx 2.24 \) meters. We decided to use this distance, as the distance measurements in camera trap distance sampling (CTDS) via reference images are assigned to intervals of 1 to 3 meters (Howe et al., 2017). We conclude this section by discussing the results independently and in relation to other existing approaches.

3.1. Robust Distance Estimation

The proposed distance estimation module consists of two parts: The DPT relative distance estimation and the alignment of said relative distance via the PVCNN. We want to emphasize again that strictly speaking, we did not develop a depth / distance estimation method, but rather trained a model to accurately align already given relative depth to match the real world metric depth without a given ground truth. Consequently, we do not compare this module to other existing depth estimation approaches. Instead, we juxtapose the PVCNN to the Random Sample Consensus (RANSAC) alignment method. We evaluate its alignment quality by comparing a transformed DPT-depth image to its corresponding ground truth image. We consider the complete depth image and the median depth value of the segmentation mask of each detection separately.

**RANSAC:** To measure the alignment performance of the PVCNN we need other alignment procedures as a point of comparison. We decided to use RANSAC (RANdom SAmple Consensus) (Fischler and Bolles, 1981) on every image to align the DPT relative disparity \( \hat{d} \) (Section 2.2) to the ground truth \( g \). To this end we invert \( g \) and estimate the unknown scale \( m^* \) and unknown shift \( c^* \) using RANSAC such that the parameters minimize the absolute disparity error:

\[
(m^*, c^*) \approx \arg \min_{m^*, c^*} |m^* \cdot \hat{d} + c^* - \frac{1}{g}|
\]  

Next, we convert our relative disparity \( \hat{d} \) to a metric depth image \( d \) using \( m^* \) and \( c^* \):

\[
d = \frac{1}{d \cdot m^* + c^*}
\]  

**Filling Gaps in Depth Data:** To select one distance value for every detection the segmentation mask of an animal (Fig. 9) is applied to the corresponding depth image and the median of their values is taken. Some animals in the groundtruth were detected at the right side of the frame, where some
depth pixels of the groundtruth had no valid depth or incorrect values (Fig. A.11 Lindenthal). To solve that problem, we calculated the DPT relative depth for that frame and aligned the DPT depth with the valid pixels in a 100x100 crop around the detection in the original depth image. The annotation mask was then applied to the aligned DPT image in the same way as previously mentioned. The RANSAC algorithm was employed to align the crop as it achieved good results while being robust to the incorrect outlier values (Table 3).

**Testing:** The PVCNN receives the approximated depth as input, while RANSAC processes the DPT disparity image and the ground truth depth (Section 2.1) as the alignment goal. Afterwards, the resulting aligned depth images of the two algorithms are compared to the ground truth Lindenthal depth images for distance values smaller than 25 meters, as this is the realistic application range for camera trap videos (Capelle et al. 2019), (Corlatti et al. 2020). The animal enclosure seen in the Lindenthal-dataset as well as the annotated animals are located in a distance smaller than 20 meters from the camera. During training of the PVCNN only 25K points are sampled for each point cloud version of a depth image to prevent overfitting and to increase robustness. For testing, we double the number of sampled points to 50K. The remaining hyper-parameters keep their training values.

One training epoch on an Intel Xeon 4215, a Nvidia P5000 and 30 GB of RAM took approximately 2.5 hours, while the loading and augmentation of the ground truth, as well as the precomputed DPT images were responsible for most of the processing time.

**Metrics:** We work with the following spatial depth metrics, which are commonly applied. $N$ denotes the total number of valid pixels, invalid pixels are masked out during evaluation. $d_i$ and $g_i$ are the estimated and ground truth depths of pixel i, respectively:

\[
\text{Mean relative error (Rel): } \frac{1}{N} \sum_{i=1}^{N} \frac{||d_i - g_i||_1}{g_i} 
\]

\[
\text{Root mean squared error (RMS): } \sqrt{\frac{1}{N} \sum_{i=1}^{N} (d_i - g_i)^2} 
\]

\[
\text{Mean Absolute Error (MAE): } \frac{1}{N} \sum_{i=1}^{N} |d_i - g_i| 
\]

\[
\text{Mean Error (ME): } \frac{1}{N} \sum_{i=1}^{N} (d_i - g_i) 
\]
**Results:** We see in Table 3 that for the complete depth image, the RANSAC algorithm outperforms the PVCNN only in the REL and MAE metric, despite having the ground truth as the alignment goal. The worse RMS and ME results can be explained by few outliers that negatively impact the overall average, whereas the PVCNN seems to yield more consistent outputs and produces less extreme distributions (Fig. 6). Additionally, the PVCNN is not far behind RANSAC, only by 19 cm in the MAE and by 0.05 in the REL. The worse relative performance for higher distances of the PVCNN can be attributed to the training loss that focuses on closer objects and the loss of detail for distant scenery caused by the downsampling of the DPT input image. Regardless of the fact that the PVCNN never saw the Lindenthal dataset during training and the alignment via RANSAC and the PVCNN are executed in two different domains, the output distributions of both approaches show a similar clustering of scales and shifts per scene. For example the clusters for scene S05 and S08 are both isolated from the other outputs and S00, S01, S03 and S04 are closely positioned to each other and display a considerable overlap. The plot demonstrates that the PVCNN indeed learned to understand the scenes and not just maps a point cloud with a given focal length to a specific scale and shift, as it displays behaviour similar to the RANSAC alignment.

|                      | RMS  | REL | MAE  | ME  |
|----------------------|------|-----|------|-----|
| Complete GT image    | 2.5695 | 0.1428 | 1.978 | -0.2322 |
| PVCNN                | 2.8772 | 0.0985 | 1.0075 | 0.4411 |
| RANSAC               |       |      |      |     |
| Instance Depth on GT BBs | 1.6821 | 0.1130 | 0.9864 | 0.1754 |
| PVCNN + DINO + Median |       |      |      |     |
| Table 3: Depth Metrics: Input as described in Section 2.1, Evaluated for distances smaller than 25 meters |

For Wild Life Distance Sampling the accuracy of depth estimation on animal instances is of utmost importance. Therefore, we additionally evaluate the PVCNN performance on the provided GT Bounding Boxes. For each detection we apply DINO (Section 2.2.4) to separate animal from background pixels and then use the median value of the corresponding depth pixels as an estimation of the animal distance. We compare this value to the ground truth distance of the animal extracted from the depth images with the median of the annotated pixel mask (Fig. 9). The metrics display an additional improvement, with an MAE of only 0.99 m, a significantly lower RMS of only 1.68 and a REL of 0.113, suggesting a higher precision of the distance estimation for closer and non-background objects.

As can be seen in figure 5, the distribution of estimated distances on the ground truth detections and of the complete pipeline indeed reflect the ground truth distribution. At 6 m the PVCNN + GT and the GT distribution differ by about 1 percentage point, while the difference at 11 m and 16 m is about 2 percentage points. Both the mean error and the mean absolute distance error measures depend strongly on the scene, as can be seen in figure 4. High estimation errors can be observed for scenes recorded during nighttime that show a part of the
Figure 4: Box plot of the distance estimation error on Ground Truth Detections per scene up to 25 m

Figure 5: Probability density of the ground truth (GT), PVCNN + DINO on GT Bounding Boxes and the complete Pipeline

roof the camera was mounted on at the top of the image (S10, S13, Fig. 8), as this object introduces a new reference point that, together with the flickering of the light, changes the complete relative scale of the image. This means that the PVCNN has to compute a new scale and shift for an almost identical looking scene which results in a high spread of possible values (Fig. 6) and consequently a higher range of estimated distances.

3.2. Detection

Since the results of the SORT 2.5D algorithm depend on the quality of the input detections determined by the MegaDetector, we complete a short evalu-
Generally the algorithm recognizes animals in unseen camera data reliably, with a mean average precision of 87.14%. Only scenes with several visible animals seem to pose a problem, like scene S04, where the model has difficulty differentiating between single subjects of a group of ducks or geese and often misses detections (False Negatives). The issue occurs in sequence S01 for goats and in sequence S09 for deer as well. Another problem seems to be the time of recording, when an animal is only barely visible during nighttime and not in the cone of light of the camera trap (S10). The amount of false alarms (FP) per sequence is mostly small, which is of utmost importance for CTDS, as false negatives can be compensated by the tracker and future correct detection in the video, while continuous FP’s are the reason for the creation of fake individuals and the distortion of the animal count.

To avoid the latter, [Beery et al., 2019] propose a post-processing step for static camera trap recording that eliminates consistent false positives, when the model incorrectly classifies parts of the background as an animal. However, that step requires manual intervention of the user and is therefore not considered for the pipeline.
### Table 4: Pascal VOC Metrics of MegaDetector on Lindenthal; True Positives (TP), False Positives (FP), False Negative (FN), Recall, Mean Average Precision (mAP), Def.: Section 3.4

| Scene | TP  | FP  | FN  | Recall | mAP  |
|-------|-----|-----|-----|--------|------|
| S01   | 203 | 5   | 54  | 79%    | 78.96% |
| S02   | 9   | 1   | 0   | 100%   | 100%  |
| S03   | 42  | 0   | 1   | 97.67% | 97.67% |
| S04   | 146 | 20  | 92  | 61.34% | 58.73% |
| S05   | 82  | 0   | 0   | 100%   | 100%  |
| S06   | 23  | 0   | 0   | 100%   | 100%  |
| S07   | 16  | 0   | 1   | 94.12% | 94.12% |
| S09   | 84  | 4   | 21  | 80%    | 79.92% |
| S10   | 39  | 1   | 43  | 47.56% | 47.23% |
| S11   | 41  | 0   | 0   | 100%   | 100%  |
| S12   | 81  | 1   | 1   | 98.78% | 98.78% |
| S13   | 37  | 0   | 4   | 90.24% | 90.24% |
| **Overall** | **766** | **32** | **216** | **78.73%** | **87.14%** |

3.3. Foreground Segmentation

The accuracy of the DINO foreground segmentation was not thoroughly evaluated due to time constraints. Sometimes the model had difficulty telling apart foreground and background pixels, especially if they had a similar color (Fig. 7 (a),(b)). RGB instead of gray scale imagery or more sophisticated contrast manipulation could possibly solve the problem. Similar difficulties occurred when the animal was at the bottom of the crop or not fully visible (Fig. 7 (c),(d)) so that the model focused on other individuals in the image. This behaviour can probably be attributed to the ImageNet training data of DINO, where the objects of interest were often in the center of the image.

![Figure 7: Problematic Cases: Differentiation Foreground / Background, Other Individuals in Wider Crop, less Focus on original detection, individual at bottom of crop](image)

3.4. Tracking

In this section we evaluate the quality of the proposed 2.5D version of SORT and apply the commonly used MOT (Multiple Object Tracking) metrics (Leal-Taixé et al., 2015). At this point of the pipeline, we have identified the animals...
in the video sequence, estimated their distance to the camera, and connected the detections to tracks based on several criteria. For each animal we have its corresponding point in 3D space and an associated track number. The MOT metrics compare the computed tracks to the given ground truth using an association metric. This means that in our 3D case, the MOT metrics assign an input 3D position to a detection of a ground truth track, if the euclidean distance for this combination in this frame is optimal and a certain threshold (here $\sqrt{5} \approx 2.24$ meters) is not exceeded. We decided to use this approximated threshold, as in traditional CTDS via reference images the distance measurements are assigned to intervals of 1 meter for 0-8 meters, to intervals of 2 meters for 8-12 meters and to an interval of 3 meters for 12-15 meters (Capelle et al., 2019). Considering that the Lindenthal dataset only includes the annotations for every tenth frame, we equally evaluate the pipeline output for every tenth image. The MOT-metrics include up to 17 different metrics; we focus on the measurements described in appendix Appendix B.

Before we start to analyse the results, we want to point out that the 3D tracks are the product of four different components that each contribute to possible errors or even reinforce wrong predictions. Therefore referencing the previous module-wise evaluations is strictly necessary for context. Additionally, isolating the effect of the tracking on the quality of the distance estimation is difficult, as the algorithm changes and drops bounding boxes if applied to ground truth detections. Therefore, we only provide the tracking evaluation on the complete pipeline.

The SORT version achieves a MOTA score of 56.3%, an average localization precision for correct detections of only 0.648 meters (MOTP) and a high precision of 90.3%. However, due to the false negatives of the MegaDetector, an overall recall of only 63.8% is reached. For CTDS, the correct number of detected objects and their accurate distances are required. In this regard the 2.5D SORT algorithm yields only 7 identity switches (IDs) and correctly recognizes 33 of 35 ground truth detections.

On a scene level, the sequences with bad MegaDetector performance (S01, S04, S10) equally display the worst tracking performance due to missing detections (FN). During the detection evaluation, only S04 showed a considerable amount of false positives (FP), whereas SORT 2.5D additionally created FPs for S01, S10, and S13. In the 3D MOT metrics a detection is recorded as a FP, if no ground truth detection is available in the area of the threshold of $\sqrt{5}$ meters, that was not yet mapped to another tracking detection. This, together with a high MOTP score, indicates an inadequate distance estimation of the individuals. One reason could be that, apart from S01, the other scenes were recorded during nighttime, where the viewing range was limited by the cone of light and thus contained no background and less depth cues for the DPT prediction and the depth alignment (Fig. 8). Especially S13 suffers from this effect, as the distances are constantly overestimated by ca. 4 meters (Fig. 4) and thus exceed the threshold of 2.25 meters.

To highlight the strengths and weaknesses of the tracking pipeline, we per-
form a qualitative evaluation of the 3D tracks calculated for S01 by the SORT 2.5D version. We use the aligned depth image and the calculated tracks to unproject the color images back to a 3D point cloud and visualize the tracks with colorful squares (Fig. 10). S01 (Fig. 9) was recorded during daytime and shows 10 goats at various distances evenly spread throughout the enclosure. One goat is not visible in the first frame, as it only enters the field of view for a short time in the middle of the video close to the camera. Two goats graze in front of a small hill and are hardly visible, as they blend in with the background. During the video the animals change their position, leave the frame, or temporarily disappear behind the hill (see groundtruth Fig. 10).

We can see that our approach yields continuous and stable tracks, that closely resemble the original. However, the output also shows several deficits: The animal at the bottom left is not recognized by the pipeline, which can be attributed to the MegaDetector and the fact that the animal is only partially visible for a short amount of time. Furthermore, the tracks of the goat that appears behind the hill and the track of the goat directly to the left of the hill are merged. The reason for that behaviour is that in the moment the goat disappears behind the hill, the goat in front of the hill is recognized by the MegaDetector and then associated with the same track, instead of generating a new one. As the upper threshold for depth association ($DIST_{max}$) in the association metric is 4.096 meters, no new track is created. What also stands out is the distortion of that track towards the front at the left and right of the hill, the reason for that being that the DPT ceases to differentiate between the goat and the slope at these points. Likewise, Figure 10 demonstrates a general problem of depth ground truth data: The track of the goat in the back shows inconsistent depth values with oscillating changes. One cause could be the insufficient resolution of the depth camera for higher distances, which leads to the merging of animal and environment pixels and an erroneous stereo depth.

The optimal strategy for both SORT versions for S09, determined by the random search (Table 2), seems to be to only consider detections with high confidence. On one hand, this minimizes the influence of false positive MegaDetector detections. On the other hand, this also discards correct detections and could lead to track fragmentation due to gaps in the detection path. To compensate for these additional FNs, the tracks are kept alive for as long as possible (high $AGE_{max}$) and require only a minimal association score (low $IOU_{min}$, low
\( SIMSCORE_{\text{min}} \). As a result of the high confidence threshold, just one detection has to be collected as evidence to create a new track \((HITS_{\text{min}} = 1)\). The SORT 2.5D version additionally incorporates a detection’s distance into the association score with a weight of \(1 - \alpha = 0.573\) and a \(DIST_{\text{max}}\) value of 4.096 meters. This gives the tracker the possibility to compensate the failure of one of the components with the value of the other one. However, the Lin- denthal dataset displays a heavily controlled scenario with clear visibility and slow movement of the individual animals. For camera traps in forest environments that capture groups of animals at a time, an adjustment of \(HITS_{\text{min}}\) and \(DIST_{\text{max}}\) might be necessary to avoid identity switches or the merging of tracks as seen in S01. Especially for CTDS only the correct number and overall positions of tracks is important, the perfect frame-by-frame localization is less relevant.

As a conclusion, we can say, that for an increase of the performance of the overall tracking, apart from the improvement of the separate modules, a more sophisticated and manual in-depth hyperparameter optimization could be a solution.
Ground truth 3D tracks

Pipeline 3D tracks

Figure 10: 3D Point Cloud Unprojection of Depth Images with Tracks (S01, c.f. figure 9) cropped to 20 meters
3.5. Discussion

Compared to previous distance estimations applied to CTDS that worked with reference material or onsite measurements, our method is fully automatic and only requires the setup of the camera trap and its focal length. The execution of the pipeline on a video of 450 frames only takes ca. 4 minutes on a setup with an Intel Xeon 4215 CPU, 30 GB of RAM and a Nvidia P5000 GPU and can be left unattended. Due to the time savings, the complete automation of the process enables the possibility of large-scale animal abundance studies and could accelerate biodiversity research. Although a direct comparison is difficult due to the different generation settings of the used datasets, the MAE per instance of 0.9864 m on single detections (Section 2.5) compared to MAE of 1.85 m of the interactive approach of [Haucke et al., 2022] may give an indication of the accuracy of the presented approach.

Moreover, it is important to point out two issues of the dataset collection and depth datasets in general: The first issue being, that outdoor stereo depth is imprecise and often inaccurate for complex environments and especially high distances. One example for this kind of error is the depth image of the UASOL dataset in Fig. A.11 in which the sky above the steel structure is assigned a distance of 10 to 13 meters. The second issue is that the datasets are not accurately cleaned, they often contain blurry color images (LVPD), completely black images due to render errors (TartanAir), or images that almost exclusively display the sky with no reference points. These deficits negatively impact the training and evaluation process and are another reason why a depth loss function that prioritizes lower distances was the sensible choice. A possible solution for the problem could be synthetic datasets generated by rendering pipelines like BlenderProc [Denninger et al., 2019] that provide diverse combinations of scenes, light, weather, camera angles, etc.

4. Conclusion

We propose a fully automated and modular processing pipeline for tracking animals and their distances in wildlife camera trapping. We derive absolute distances in metric values based on monocular relative distance estimation by exploiting a novel 3D point cloud-based alignment model that is trained on a diverse collection of outdoor datasets. We detect and localize animals using our multi-instance DINO method. We test the optimized approach in a zero-shot evaluation on an unseen zoo scenario dataset and achieve a mean absolute error of only 1.1978 meters over the entire image. For the abundance estimation of unmarked animal populations, the correct number and distances of animals are important for an accurate estimation. Applied to the zoo scenario dataset, our pipeline correctly registers 33 of 35 animals with an MAE (Mean Absolute Error) of 0.648 meters for correct classifications, with a recall of 63.8% and a precision of 90.3%. But most importantly, our approach simplifies and accelerates the overall workflow of biodiversity researchers considerably by relieving
them from the demand of generating reference data including time-consuming manual preparation, interaction and post-processing of these data material.
Acknowledgement

This work is partially funded by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung (BMBF), Bonn, Germany (AMMOD - Automated Multisensor Stations for Monitoring of BioDiversity: FKZ 01DK17048). This funding is gratefully acknowledged.

We thank Thomas Ensch, Michael Gehlen and the entire team of the Lindenthaler Tierpark for their cooperation by hosting the experimental camera trap hardware on-site. We thank Alejandro Berni Garcia for his help with the construction of the wooden camera trap casing.
### Appendix A. Data

| Scene | Time   | Animals       | Annotated |
|-------|--------|---------------|-----------|
| 00    | Day    | Geese, Ducks  | No        |
| 01    | Day    | 9 Goats       | Yes       |
| 02    | Day    | 2 Donkeys     | Yes       |
| 03    | Day    | 1 Deer        | Yes       |
| 04    | Day    | 6 Geese       | Yes       |
| 05    | Night  | 2 Deer        | Yes       |
| 06    | Evening| 2 Deer        | Yes       |
| 07    | Night  | 1 Deer        | Yes       |
| 08    | Day    | None          | No        |
| 09    | Day    | 5 Deer        | Yes       |
| 10    | Night  | 2 Deer        | Yes       |
| 11    | Night  | 1 Deer        | Yes       |
| 12    | Night  | 2 Deer        | Yes       |
| 13    | Night  | 1 Deer        | Yes       |

*Table A.5: Lindenthal Scene Overview*
Figure A.11: Datasets Image Overview: RGB with corresponding depth image and scale. White pixels are invalid and are masked out during training and evaluation.
Appendix B. Tracking Metrics

- **GT**: Number of ground truth objects in this sequence
- **NT** (*NT* \(\equiv\) *GT*): Number of different detected objects in this sequence
- **FP** (\(\downarrow\)): Number of false positive detection
- **FN** (\(\downarrow\)): Number of false negatives / missed detection
- **FM** (\(\downarrow\)): Number of track fragmentations / interruptions of a ground truth trajectory, where the track is interrupted by a missing detection
- **IDs** (\(\downarrow\)): Number of identity switches
- **MT** (\(\uparrow\)): Number of mostly tracked trajectories, i.e. the target has the same label for at least 80% of its lifespan
- **PT** (\(\uparrow\)): Number of objects tracked between 20 and 80 percent of lifespan
- **Recall** (\(\uparrow\)): \(\frac{TP}{TP+FN}\), “general ability of model to identify relevant objects”
- **Precision** (\(\uparrow\)): \(\frac{TP}{TP+FP}\), “ability of model to only identify relevant objects”

- **MOTA** (\(\uparrow\)): MOT Accuracy, \(MOTA = 1 - \frac{\sum_t(FN_t+FP_t+IDs_t)}{\sum_tGT_t}\), where \(t\) is the frame index, MOTA is “the number of errors over the number of objects in the scene”

- **MOTP** (\(\uparrow\)): MOT Precision, the average dissimilarity between true positives and corresponding ground truth, e.g. localization precision. \(MOTP = \frac{\sum_t \sum_i d_{t,i}}{\sum_t c_t}\), where \(c_t\) denotes the number of matches in frame \(t\) and \(d_{t,i}\) is the absolute euclidean distance of target \(i\) to its assigned ground truth object.

Appendix C. Tracking Results

For every input sequence of frames, SORT manages a set of currently active trackers. Each tracker \(X\) has a state vector:

\[ X = [u, v, s, r, u', v', s'] \]  \hspace{1cm} (C.1)

\(u\) and \(v\) represent the horizontal and vertical pixel location of the centre of the detection, whereas \(s\) represents the scale (area) and \(r\) represents the detection’s bounding box aspect ratio. The aspect ratio is assumed to be constant. The remaining variables describe the complementary velocity. For each frame, the steps in Algorithm [1] are executed.
Algorithm 1: SORT per Frame

Data: active trackers, frame detections
Result: updated trackers, output

1. forall trackers X in active trackers do
2.   change state of X to predicted state via Kalman filter;
3. end

4. forall possible combinations of active tracker X, detection D do
5.   if confidence of D > CONF_{min} then
6.     calculate association score assoc(X,D) of X and D;
7.   end
8. end

9. solve optimal assignment of assoc(X,D) via Hungarian Algorithm;

10. forall optimal assignments (X,D) do
11.   if association score of (X,D) < IoU_{min} then
12.     discard assignment (X,D)
13.   end
14. end

15. forall detections D not assigned to a tracker do
16.   add new tracker for D to active trackers;
17. end

18. forall trackers X in active trackers do
19.   Update X with its detection assignment via Kalman framework;
20.   if X has not been updated since AGE_{max} frames and has not
21.     received an assignment then
22.     delete X from active trackers;
23.   end
24.   if X has been updated ≥ HITS_{min} times then
25.     add state of X to output;
26.   end
27. end

28. return active trackers, output
| Scene | Recall | Prec | GT  | NT  | MT  | PT  | ML  | FP  | FN  | IDs | FM  | MOTA | MOTP |
|-------|--------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|
| S01   | 59.5%  | 87.9%| 10  | 9   | 3   | 6   | 1   | 21  | 101 | 5   | 12  | 49.4%| 0.610|
| S02   | 77.8%  | 77.8%| 2   | 2   | 1   | 0   | 1   | 2   | 2   | 0   | 0   | 55.6%| 1.017|
| S03   | 86.0%  | 90.2%| 1   | 1   | 1   | 0   | 0   | 4   | 6   | 0   | 4   | 76.7%| 1.411|
| S04   | 46.6%  | 100.0%| 6   | 5   | 0   | 5   | 1   | 0   | 127 | 1   | 26  | 46.2%| 0.782|
| S05   | 98.8%  | 98.8%| 2   | 2   | 2   | 0   | 0   | 1   | 1   | 0   | 0   | 97.6%| 0.517|
| S06   | 100.0%| 100.0%| 2   | 2   | 2   | 0   | 0   | 0   | 0   | 0   | 0   | 100.0%| 0.380|
| S07   | 94.1%  | 100.0%| 1   | 1   | 1   | 0   | 0   | 0   | 1   | 0   | 0   | 94.1%| 0.276|
| S09 Hyper | 78.1% | 98.8%| 5   | 5   | 4   | 1   | 0   | 1   | 23  | 1   | 9   | 76.2%| 0.449|
| S10   | 20.7%  | 70.8%| 2   | 2   | 0   | 1   | 1   | 7   | 6   | 0   | 0   | 12.2%| 0.965|
| S11   | 100.0%| 100.0%| 1   | 1   | 1   | 0   | 0   | 0   | 0   | 0   | 0   | 100.0%| 0.342|
| S12   | 95.1%  | 96.3%| 2   | 2   | 2   | 0   | 0   | 3   | 4   | 0   | 3   | 91.5%| 0.697|
| S13   | 12.2%  | 13.9%| 1   | 1   | 0   | 0   | 1   | 31  | 36  | 0   | 1   | -63.4%| 1.476|
| Overall| 63.8%  | 90.1%| 35  | 33  | 14  | 13  | 5   | 70  | 369 | 7   | 55  | 56.3%| 0.645|

Table C.6: SORT 2.5D MOT Metrics on Lindenthal 3D Positions. Association Score: Euclidean Distance of max. 2.24 meters
References

Bauer, Z., Gomez-Donoso, F., Cruz, E., Orts, S., and Cazorla, M. (2019). Uasol, a large-scale high-resolution outdoor stereo dataset. Scientific Data, 6:1–14.

Beery, S., Morris, D., and Yang, S. (2019). Efficient pipeline for camera trap image review. arXiv preprint arXiv:1907.06772.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. In 2016 IEEE International Conference on Image Processing (ICIP), pages 3464–3468.

Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). End to end learning for self-driving cars. CoRR, abs/1604.07316.

Capelle, N., Despres-Einspenner, M.-L., Howe, E. J., Boesch, C., and Kühl, H. S. (2019). Validating camera trap distance sampling for chimpanzees. American Journal of Primatology, 81(3).

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). Emerging properties in self-supervised vision transformers. arXiv preprint arXiv:2104.14294.

Cho, J., Min, D., Kim, Y., and Sohn, K. (2021). Deep monocular depth estimation leveraging a large-scale outdoor stereo dataset. Expert Systems with Applications, 178:114877.

Corlatti, L., Sivieri, S., Sudolska, B., Giacomelli, S., and Pedrotti, L. (2020). A field test of unconventional camera trap distance sampling to estimate abundance of marmot populations. Wildlife Biology, 2020(4).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255.

Denninger, M., Sundermeyer, M., Winkelbauer, D., Zidan, Y., Olefir, D., Elbadrawy, M., Lodhi, A., and Katam, H. (2019). Blenderproc. arXiv preprint arXiv:1911.01911.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM, 24(6):381–395.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. International Journal of Robotics Research, 32(11):1231–1237.
Haucke, T., Kühl, H. S., Hoyer, J., and Steinhage, V. (2022). Overcoming the distance estimation bottleneck in estimating animal abundance with camera traps. Ecological Informatics, 68:101536.

Haucke, T. and Steinhage, V. (2021). Exploiting depth information for wildlife monitoring. CoRR, abs/2102.05607.

Hofmeester, T. R., Rowcliffe, J. M., and Jansen, P. A. (2017). A simple method for estimating the effective detection distance of camera traps. Remote Sensing in Ecology and Conservation, 3(2):81–89.

Howe, E. J., Buckland, S. T., Despres-Einspenner, M.-L., and Kühl, H. S. (2017). Distance sampling with camera traps. Methods in Ecology and Evolution, 8(11):1558–1565.

Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. New Phytologist, 11(2):37–50.

Kopf, J., Matzen, K., Alsisan, S., Quigley, O., Ge, F., Chong, Y., Patterson, J., Frahm, J.-M., Wu, S., Yu, M., Zhang, P., He, Z., Vajda, P., Saraf, A., and Cohen, M. (2020). One shot 3d photography. ACM, 39(4).

Leal-Taixé, L., Milan, A., Reid, I., Roth, S., and Schindler, K. (2015). MOTChallenge 2015: Towards a benchmark for multi-target tracking. arXiv:1504.01942 [cs]. arXiv: 1504.01942.

Liu, Z., Tang, H., Lin, Y., and Han, S. (2019). Point-voxel cnn for efficient 3d deep learning. In Advances in Neural Information Processing Systems.

Moeller, A. K., Lukacs, P. M., and Horne, J. S. (2018). Three novel methods to estimate abundance of unmarked animals using remote cameras. Ecosphere, 9(8):e02331.

Moon, G., Chang, J. Y., and Lee, K. M. (2017). V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. CoRR, abs/1711.07399.

Mueller, E. and Chi, A. (2022). End-to-End Models for Complex AI Tasks. https://www.capitalone.com/tech/machine-learning/pros-and-cons-of-end-to-end-models Accessed: 2022-02-04.

Nakashima, Y., Fukasawa, K., and Sanejima, H. (2018). Estimating animal density without individual recognition using information derivable exclusively from camera traps. Journal of Applied Ecology, 55(2):735–744.

Niu, Tarapore, and Zauner (2020). Low viewpoint forest depth dataset for sparse rover swarms. The 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2020).

O’Connell, A. F., Nichols, J. D., and Karanth, K. U. (2011). Camera traps in animal ecology: methods and analyses, volume 271. Springer.
Padilla, R., Passos, W. L., Dias, T. L. B., Netto, S. L., and da Silva, E. A. B. (2021). A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics, 10(3).

Ranftl, R., Bochkovskiy, A., and Koltun, V. (2021). Vision transformers for dense prediction. CoRR, abs/2103.13413.

Rowcliffe, J. M., Field, J., Turvey, S. T., and Carbone, C. (2008). Estimating animal density using camera traps without the need for individual recognition. Journal of Applied Ecology, 45(4):1228–1236.

Rowcliffe, M., Carbone, C., Jansen, P., Kays, R., and Kranstauber, B. (2011). Quantifying the sensitivity of camera traps: An adapted distance sampling approach. Methods in Ecology and Evolution, 2:464 – 476.

Wang, D., Wang, X., and Lv, S. (2019). An overview of end-to-end automatic speech recognition. Symmetry, 11(8).

Wang, W., Zhu, D., Wang, X., Hu, Y., Qiu, Y., Wang, C., Hu, Y., Kapoor, A., and Scherer, S. (2020). Tartanair: A dataset to push the limits of visual slam. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Woo, W., Lee, W. W., and Park, N. (2011). Depth-assisted real-time 3d object detection for augmented reality. In Proceedings of the International Conference on Artificial Intelligence (ICAT), pages 2:126–132.

Yin, W., Zhang, J., Wang, O., Niklaus, S., Mai, L., Chen, S., and Shen, C. (2021). Learning to recover 3d scene shape from a single image. In Proc. IEEE Conf. Comp. Vis. Patt. Recogn. (CVPR).

Yu, Z., Macbeth, S., Modi, K., and Pujol, J. M. (2016). Tracking the trackers. In Proceedings of the 25th International Conference on World Wide Web, WWW '16, page 121–132, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.