Hochschule Darmstadt

Fachbereich Mathematik und
Naturwissenschaften
&
Fachbereich Informatik

When Segment and Track Anything Meets Wildlife Videos

Thesis for the Award of the Academic Degree
Master of Science (M. Sc.)
in the Study Program Data Science

submitted by:

Huiyi Wang

First supervisor:	:	Prof. Dr. Andreas Weinmann
Second supervisor:	:	Prof. Dr. Elke Hergenröther

Issue date: 31.07.2023
Submission date: 19.12.2023

DECLARATION

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 19. Dezember 2023

Huiyi Wang

ABSTRACT

Wildlife conservation is more important than ever to protect biodiversity and keep the balance of the ecosystem. In recent years, machine learning and deep learning have been spreading in the computer vision field and gained huge success. The advancements in this field have also made contributions to research on wildlife biology. However, considering the fact that wildlife usually lives in complex nature surroundings, research on wildlife biology still confronts a number of challenges. For instance, the identification of wildlife could be disturbed by its habitat. To reduce the effect of living surroundings and ensure that the further analysis can focus on the animals of interest, an effective strategy could be foreground/background segmentation.

Background subtraction is one of the traditional segmentation techniques and includes temporal median filter as well as statistical background modeling. These approaches might perform well in relatively simple scenarios, but they also have many constraints. Deep learning segmentation models have been proven to be effective in previous research, especially the Mask R-CNN of two-stage segmentation model and the YOLACT of one-stage segmentation model, as well as their variants. On the other hand, it requires sufficient ground-truth data of wildlife segmentation to train these models, which poses challenges in view of the limited datasets.

In this thesis, a framework is developed as a tool for automatically segmenting wildlife in video sequences that is not limited to only a few certain species. It contains mainly three components, i.e., a YOLOV5-based detection model "MegaDetector", a foundation segmentation model Segment Anything Model (SAM), and a Video Object Segmentation (VOS) model "Cutie". In addition, a matching procedure and post-processing were implemented to overcome the issue of multiple overlapping animals in video sequences. As both SAM and MegaDetector were trained with extensive datasets, they demonstrate outstanding performance by general segmentation tasks and wildlife detection tasks, and thus the framework directly employed their pre-trained models without fine-tuning and domain adaption.

The framework was tested quantitatively with five high-resolution leopard video clips from the Pan African Programme and achieved a score (Mask IoU between predicted masks and ground-truth masks) of over 85%. Moreover, the framework was tested qualitatively with two YouTube low-resolution videos, which contain multiple overlapping animals. The results are reliable in the majority of cases.

Keywords: Wildlife Conservation, Automatic Segmentation, Foundation Segmentation Model, Video Object Segmentation

ZUSAMMENFASSUNG

Der Schutz von Wildtieren ist heutzutage wichtiger denn je, um die biologische Vielfalt zu schützen und das Ökosystem im Gleichgewicht zu halten. In den letzten Jahren haben Machine Learning und Deep Learning im Bereich der Computer Vision Verbreitung gefunden und dort Erfolge erzielt. Die Fortschritte in diesem Feld haben gleichzeitig auch einen Beitrag zur Wildtierforschung geleistet. Gleichwohl bestehen durch die Tatsache, dass Wildtiere üblicherweise in komplexen Naturumgebungen leben, nach wie vor einige Herausforderungen, mit denen sich die Wildtierbiologie konfrontiert sieht. Beispielsweise könnte etwa die Identifikation von Wildtieren durch ihren Lebensraum gestört werden. Um den Effekt der lebenden Umgebung zu reduzieren und sicherzustellen, dass der Fokus der weiteren Analyse auf den sich im Mittelpunkt des Interesses befindlichen Tieren befindet, könnte Vordergrund/Hintergrund-Segmentierung eine wirksame Strategie sein.

Die Hintergrundsubtraktion gehört zu den traditionellen Segmentierungstechniken und beinhaltet den temporalen Medianfilter sowie die statistische Hintergrundmodellierung. Diese Ansätze könnten eine gute Performanz in relativ einfachen Szenarien aufweisen, wobei sie jedoch auch vielen Einschränkungen unterliegen. Deep Learning Segmentierungsmodelle haben sich in der bisherigen Forschung als effektiv erwiesen, was insbesondere für das Mask R-CNN Modell der zweistufigen Segmentierung und das YOLACT Modell der einstufigen Segmentierung sowie deren Variationen gilt. Andererseits sind ausreichende Ground-Truth-Daten über Wildtiersegmentierung erforderlich, um diese Modelle zu trainieren, was angesichts der begrenzten Datensätze eine Herausforderung darstellt.

Im Rahmen dieser Thesis wird ein Framework als Werkzeug zur automatischen Segmentierung von Wildtieren in Videosequenzen entwickelt, welches sich nicht nur auf einige wenige bestimmte Tierarten beschränkt. Es enthält vornehmlich drei Komponenten, nämlich ein auf YOLOV5 basierendes Detektionsmodell "MegaDetector", ein grundlegendes Segmentierungsmodell "Segment Anything Model" (SAM) sowie ein Video Object Segmentation (VOS) Modell "Cutie". Außerdem wurde ein Matching-Verfahren und eine Nachbearbeitung implementiert, um das Problem von sich mehrfach überlappenden Tieren in Videosequenzen zu lösen. Da sowohl SAM als auch MegaDetector mit umfangreichen Datensätzen trainiert wurden, weisen sie eine herausragende Performanz bei allgemeinen Segmentierungsaufgaben und Wildtierdetektierungsaufgaben auf, weshalb ihre vortrainierten Modelle ohne Fine-tuning und Domänenanpassung direkt im Framework eingesetzt werden.

Das Framework wurde quantitativ mit fünf hochauflösenden Leoparden-Videoclips des Pan African Programme getestet und erreichte einen Score (Masken-IoU zwischen vorhergesagten Masken und Ground-Truth-Masken)

von über 85%. Zusätzlich ist das Framework qualitativ mit zwei niedrigauflösenden YouTube-Videos getestet worden, die mehrere überlappende Wildtiere enthalten. In den meisten Fällen sind die Ergebnisse reliable.

Schlagwörter: Wildtierschutz, Automatische Segmentierung, Grundlegendes Segmentierungsmodell, Video Objekt Segmentierung

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABKÜRZUNGSVERZEICHNIS

BLEU Bilingual Evaluation Understudy

CNNs Convolutional Neural Networks

GMM Gaussian Mixture Model

IPBES Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services

KDE Kernel Density Estimation

LSTM Long Short-Term Memory

MAE Masked AutoEncoders

Mask IoU Mask Intersection-over-Union

PDF Probability Density Function

RNNs Recurrent Neural Networks

RoI Region of Interest

SAM Segment Anything Model

SAMTrack Segment and Track Anything

TAM Track Anything Model

TDeva Tracking with DEVA

TMF Temporal Median Filter

ViT Vision Transformer

ViTs Vision Transformers

VOS Video Object Segmentation

VOT Video Object Tracking

Part I

THESIS

1 INTRODUCTION

1.1 MOTIVATION

The rapidly rising loss of various species is becoming one of the most concerning aspects of the ongoing biodiversity and ecosystems crisis[15]. According to the assessment of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES)[10], millions of animal species, including marine mammals, vertebrates, and marine fish, are currently in danger of extinction. Addressing this issue necessitates an urgent focus on effectively identifying and tracking threatened species[40].

In recent years, there has been a prominent increase in academic interest within the ecological community to implement applications for detecting, identifying, and tracking animals[70]. Since machine learning and deep learning have already pervaded the field of computer vision, there has been significant progress in research aiming to identify target[96].

However, the application of computer vision techniques in the field of wildlife research still confronts a number of challenges, which primarily arise from the complex and dynamic nature of the habitat. For instance, an erroneous matching between features extracted from the complex background and those derived from the actually observed wildlife can lead to incorrect pattern recolonization and misidentification of wildlife animals[78]. Foreground/background segmentation could be employed as a needed strategy to navigate this challenge. An appropriate segmentation framework might successfully isolate wildlife from its surroundings, lessening the impact of the background, so that the subsequent analysis could be focused on the specific animals of interest.

1.2 AIM OF THIS THESIS

Numerous efforts have been made for foreground/background segmentation. Background subtraction is a frequently employed technique for segmenting objects of interest in motion within video footages[69]. The conventional methods for background modeling encompass a spectrum of techniques, spanning from straightforward approaches, such as temporal median filter, to more advanced statistical methods, e.g., ViBe[6], SubSence[77][69].

Besides the aforementioned conventional methods, deep learning networks, because of their significant advantages, have found widespread application in computer vision tasks, such as object segmentation[87][37]. Particularly noteworthy was the introduction of the Vision Transformer (ViT), which is an epoch-making model and demonstrates vast potential for tasks involving detection and segmentation[33].

With ViT as its backbone network, a groundbreaking foundational model for general image segmentation tasks, known as the SAM, has been recently released. SAM was trained on the most extensive segmentation dataset to date, encompassing 11 million images and 1 billion masks. Remarkably, this model has proved to have outstanding zero-shot performance, even on images and segmentation tasks that have never been seen previously[51]. Encouraged by the remarkable segmentation performance of SAM, since the release, SAM has been utilized in various fields, including medical image segmentation or remote sensing image segmentation[41] [84]. However, up to the present, SAM has not been applied in wildlife research.

The primary aim of this thesis is to develop a pipeline combining SAM together with an object detection model and an object tracking model in order to perform an automatic segmentation of wildlife from its background in videos. The pipeline is seeking to address those previously already mentioned challenges in wildlife analysis, such as incorrect pattern recognition and mis-identification of wildlife, which often arise from background disruption.

The next objective is to evaluate whether this framework works reliable for complex scenarios as well as low-resolution videos or not, and how well the framework performs in such contexts.

In this thesis, the temporal median filter, one of the most commonly adopted straightforward techniques for background subtraction[42], serves as the baseline approach. In addition to that, the performance of the proposed framework is compared with the previous Video Object Tracking (VOT) frameworks.

1.3 STRUCTURE

This thesis seeks to propose a framework that combines a detection model, a segmentation model, and a VOS model for an automatic segmentation of wildlife in videos. The proposed framework is evaluated in different scenarios.

The main scenarios of this thesis include the segmentation of a single leopard in a complex environment of high-resolution (1920x1080) videos as well as the segmentation of multiple wildlife in low-resolution videos (480x360). The framework applied MegaDetector as the detector, SAM as the segmentor, and Cutie as the tracker.

The remaining part of this thesis is structured as follows: Section 2 reviews the theoretical background for segmentation as well as related previous works. Section 3 describes the sample data and the framework in detail. Section 4 presents both quantitative and qualitative results of the experiments and shows the failure cases. Finally, in the last section, after drawing conclusions from the thesis, a brief insight into future perspectives is given and possible research areas are discussed.

2

THEORETICAL BACKGROUND

2.1 BACKGROUND SUBTRACTION FOR SEGMENTATION

Background subtraction is one of the most widely adopted techniques for segmenting objects of interest, commonly referred to as "foreground"[79]. The origin of background subtraction can be traced back to the late 1970s when Jain and Nagel[45] released their groundbreaking work on background subtraction. Their approach is to detect moving objects by analyzing the difference between consecutive frames[45][76]. Throughout the years, various background subtraction methods have emerged, from straightforward approaches, such as the temporal median filter, to more advanced statistical methods, for example, ViBe[6] and SubSence[77][69]. ViBe[6] and SubSence[77][69] have proven their excellent performance, which is evident in the superior F-Measure score when compared to other statistical approaches on the CDnet2014 datasets[11].

The underlying principle of background subtraction involves comparing the observed frame with an estimated reference frame, also called "background model", which is expected to contain no objects of interest [69]. Consequently, the result of the subtraction between the observed frame and the reference frame represents the objects of interest ("foreground")[56]. This process can be described with the following formula 2.1(1). $Mask_t(x,y)$ represents a binary mask value of a pixel (x,y) at time t, indicating whether the pixel (x,y) belongs to a moving object or not, $B(x,y)$ is a background model for the pixel at the location (x,y), $I_t(x,y)$ denotes the pixel value at time t located at (x,y), the value d represents the distance between $I_t(x,y)$ and $B(x,y)$, τ stands for the the threshold used for object segmentation[75].

$$Mask_t(x,y) = \begin{cases} 1 & if\ d(I_t(x,y), B(x,y)) > \tau \\ 0 & otherwise \end{cases} \tag{2.1(1)}$$

2.1.1 *Temporal Median Filter*

As a frequently used background modeling technique, TMF works based on the assumption that every pixel holds the background value across over half of the entire video frames[54]. In other words, a pixel is assumed to be part of the background for a longer duration in a video than it is segmented as part of the foreground[58]. TMF calculates the median of past frames or the sub-sampled previous frames as the estimated background [28][42]:

$$B(x,y)_K = Median(I(x,y)_{K-\Delta k}, I(x,y)_{K-2\Delta k} \ldots I(x,y)_{K-n\Delta k}) \tag{2.1(2)}$$

In equation 2.1(2), $I(x,y)_{K-\Delta k}$, $I(x,y)_{K-2\Delta k}$,..., $I(x,y)_{K-n\Delta k}$ represents pixel values situated at (x, y) over the frames, which are selected at a rate of one every Δk from the previous frames of the current Kth frame, for the purpose of building the background $B(x,y)_K$. Figure 2.1 demonstrates a background estimation using TMF.

(a) TMF: selected input frames for background modeling

(b) TMF: estimated background as the median of the selected frames

Figure 2.1: An Example of Background Modeling Using TMF

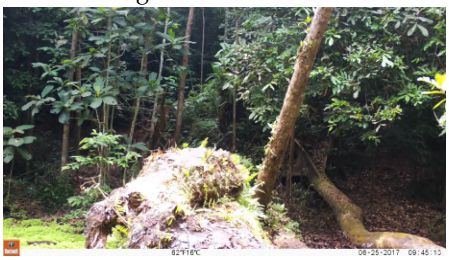
TMF has proved its effectiveness in previous research. In the study of Lo and Velastin[55], the authors have applied the temporal median technique to obtain a precise background, ensuring robust performance for their developed "Automatic Congestion Detection System on Underground Platforms". Cucchiara [28] highlighted in their research that temporal median filter is able to robustly build the background with low computational cost. Hung[43] proposed a median repetition checking algorithm that not only accelerates the speed of the temporal filter but also lowers computational expenses, thus making it more suitable for practical applications.

However, despite its effectiveness and advantages, the temporal median filter has several limitations, one of them being an inevitable result of its underlying assumption. As mentioned above, the temporal median filter works only on the assumption that the background must be observable in more than 50% of the total time across the video frames[58]. Otherwise, the filter may mistakenly take parts of the foreground as background. A failed exam-

ple for estimating the background using TMF is shown in Figure 2.2. An additional disadvantage directly linked to the temporal median filter is its storage demand, which is caused by the requirement to maintain a buffer for storing selected frames used in the estimation of the reference background[69]. Furthermore, employing the TMF for background estimation requires a background that remains consistently stable and thus poses challenges in achieving adaptive background estimation for dynamic or noisy environments[54].

(a): Three selected input frames for the background estimation

(b): Failed estimated background



(c): Expected estimated background

Figure 2.2: An Illustration of Failed Background Modeling Using the TMF.

2.1.2 *Statistical Methods for Background Subtraction*

Environments are usually dynamically changing in the real world. Because of the inherent complexity and variety of real-world scenarios, such as scenes where trees often move in the wind, relying purely on median-based techniques for background subtraction might become difficult since acquiring an accurate and reliable static reference image from frames with dynamic background seems quite challenging, if not impossible[6].

To deal with this issue caused by non-static background, various background modeling techniques have been investigated. These background modeling techniques can be mainly categorized into two major groups: those based on Probability Density Function (PDF) and those sample-based approaches that are directly based on truly observed samples[80].

Representative approaches of the first group are Gaussian Mixture Model (GMM)[98] and Kernel Density Estimation (KDE)[34]. GMM assumes that pixels can be seen as a result of multiple Gaussian distributions, each with different weight. To fit an appropriate background model, it is necessary to update not just the parameters (mean and variance) of each Gaussian distribution but also to dynamically adjust the number of distributions as well as their associated weights[98]. In cases where the complex background cannot be adequately described by parametric models like GMM, non-parametric models such as KDE have been proposed[14][34]. As highlighted in the work

of Elgammal et al.[34], KDE abandons employing specific parametric distributions and instead estimates the underlying probability density function using a kernel function, enabling a more flexible and accurate modeling of the dynamic scenes[99].

However, statistically, the accuracy and robustness of any distribution estimation, including statistical background estimation in computer vision, are dependent on the amount of data employed for estimation, implying that the initialization for estimating the pixel distributions should use sufficient video frames to ensure a reliable model[6]. In order to minimize the data needed for initialization and accelerate the modeling process, alternative sample-based approaches have been proposed, such as "VIsual Background Extractor" (ViBe)[6] and "Self-Balanced SENsitivity SEgmenter" (SubSENSE)[77], which only require a single frame for initialization.

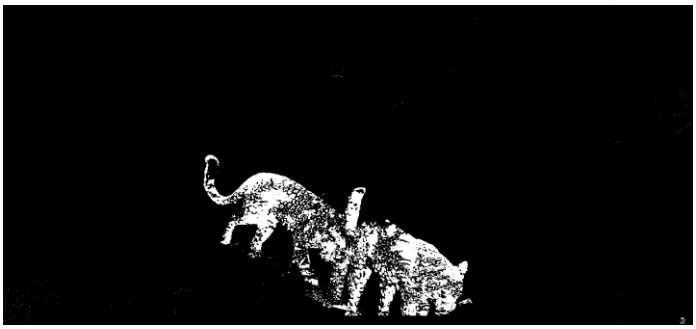
In both methods, each pixel x has its background model represented by a set of background samples $\{v_1, v_2, \ldots, v_N\}$ that were truly observed from previous frames. The background model of each pixel is initialized by selecting its neighboring pixels $N_G(x)$ randomly as background samples[6][77]. Whether the pixel x belongs to background pixels or foreground pixels depends on the distance between its current pixel value and its previously saved background samples. If the pixel x is classified as a background pixel at the time t, it could be used for updating the background samples in the pixel model $M_t(x)$. By continually updating the background samples for each pixel, ViBe and SubSENSE are able to adapt to dynamic background changes.

$$M(x) = \{v_1, v_2, \ldots, v_N\} \qquad M^0(x) = \{v^0(y|y \in N_G(x))\} \tag{2.1(3)}$$

However, the authors noted that the initialization works on the assumption that the first frame must be free of any objects of interest to avoid generating ghost area; otherwise the ghost area might be only gradually disappeared after a long period[6][77]. Despite the better performance in comparison to alternative statistical methods[11], ViBe and SubSENSE still struggle with the challenge of eliminating ghost areas, as shown in Figure 2.3.

2.2 DEEP LEARNING FOR SEGMENTATION

As previously noted, conventional segmentation techniques, such as the aforementioned TMF and the statistical background subtraction methods, are efficient in many practical applications. However, their performance is constrained by their underlying statistical assumptions. In recent years, deep learning networks from CNNs to transformers have gained widespread application for object-segmentation tasks[87][37]. Especially the invention of the transformer lead to a revolutionary change in the field of computer vision[81].



(a) Ghost area problem using ViBe

(b) Reason for ghost area: the first frame contains the target of interest

Figure 2.3: Ghost Area Problem

2.2.1 *Basic Transformer*

- Encoder-Decoder Architecture

The encoder is a network that converts an input sequence $x = (x_1, x_2....x_n)$ into a fixed-length vector sequence $z = (z_1, z_2....z_n)$. The decoder takes the vector sequence z as input and produces the final output sequence $y = (y_1, y_2....y_m)$[5][81]. The length m of the output sequence y can differ from the length n of the input sequence x since the use of the EOS (end-of-sequence) token enables the decoder to generate sequences of flexible length[66].

The encoder-decoder framework can incorporate different kind of neural networks. Cho et al.[23] propose using Recurrent Neural Networks (RNNs) in both encoder and decoder in their model "RNN Encoder-Decoder" for translation tasks, and their model shows impressive performance in terms of Bilingual Evaluation Understudy (BLEU)[74] scores. Moreover, depending on the specific requirements of the task, other types of neural networks, such as Long Short-Term Memory (LSTM) and CNNs, are also frequently applied within the encoder-decoder framework[57][83].

Vaswani et al.[81] presented in their work "Attention Is All You Need" a groundbreaking model known as the "Transformer", which is a revolutionary encoder-decoder model that completely relies on attention mechanism. The basic structure of transformer is illustrated in Figure 2.4.

The left side of Figure 2.4 shows the structure of the encoder within the transformer architecture. The encoder is made up of N identical blocks, each

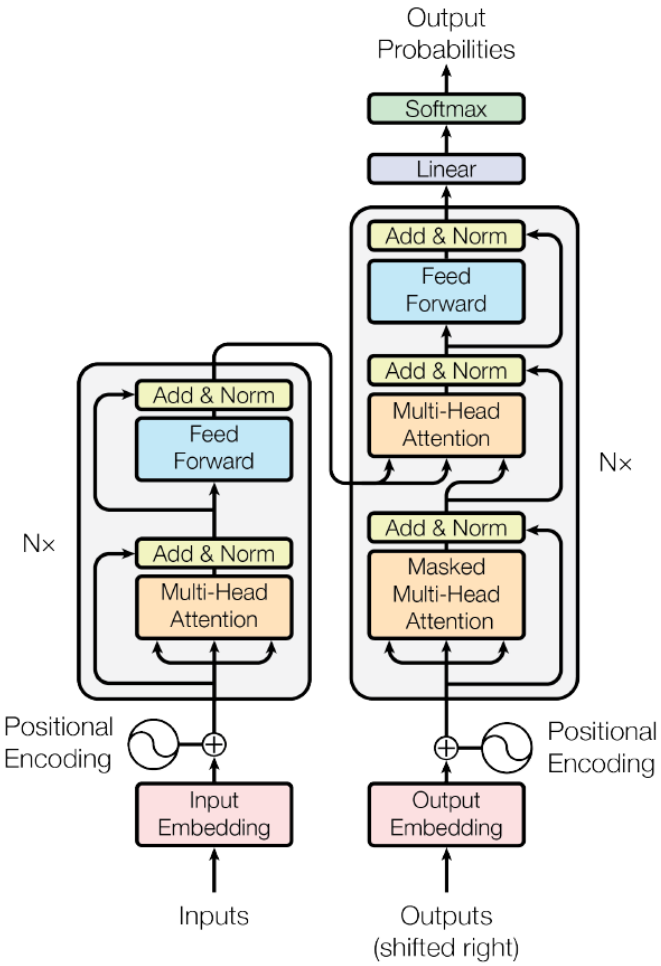


Figure 2.4: Transformer Structure[81]

with two primary layers: a multi-head attention mechanism and a fully connected feed-forward network. After each layer, a residual connection and a layer normalization are attached.

The structure of the decoder, as shown on the right side of Figure 2.4, also comprises N identical blocks. A notable difference to the attention layer in the encoder is that the attention layer in the decoder is adapted into a masked self-attention layer since the decoder is an auto-regressive (AR) model, in which the previously generated token serves as input for the generation of the next token. Besides the masked self-attention layer and the fully connected feed-forward network, a cross-attention mechanism is added in the decoder as a bridge between encoder and decoder, allowing the decoder to utilize the output of the encoder.

- Attention Mechanism

As already mentioned, the application of the attention mechanism, which attempts to generate the final context vectors by capturing contextual information from the input sequences, is the highlight of transformer. Figure 2.5

and Figure 2.6 illustrate in detail the process of how a single-head attention mechanism transforms the input vectors a=(a^1,a^2,a^3) into the output vectors b=(b^1,b^2,b^3), each element of the output vector b containing information from all elements of the input vectors. a is the input vectors of a dimension d_{model}, originating from an input sequence that consists of n tokens. Each input vector is linearly projected into a query vector as well as a key vector of dimension d_k, and a value vector of dimension d_v. Mathematically, for instance, q^1 =a^1*W^q, k^1 =a^1*W^k, v^1 =a^1*W^v. W^q, W^k, W^v are the learnable weight parameters obtained from training[81].

Once Q, K, V have been obtained, scaled dot-product attention calculates the dot products of query Q - key K pairs and scales these products by dividing them with a predefined factor $\sqrt{d_k}$ to hinder the vanishing gradients issue that can arise when the softmax function is applied since very large dot products would result in extremely small gradients within the softmax layer, leading to the vanishing gradient problem. After applying the softmax function, attention weights α' are generated, demonstrating the contribution of each input element to the final context vector b, which is the weighted sum based on the attention weights α' across all elements of the value vectors V[1]. Self-attention works with queries Q, keys K, and values V generated within the same sequence, whereas cross-attention serves as a connection between the encoder and the decoder since it uses keys K and values V generated by the encoder and queries Q generated by the Decoder[81].

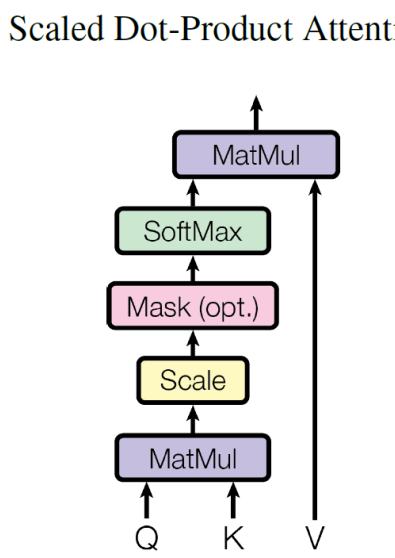


Figure 2.5: Scaled Dot-Product Attention[81]

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.2(1)}$$

with $Q = W^q I$, $K = W^k I$, $V = W^v I$, I is the input. W^q, W^k,W^v are the metrics to be learned.

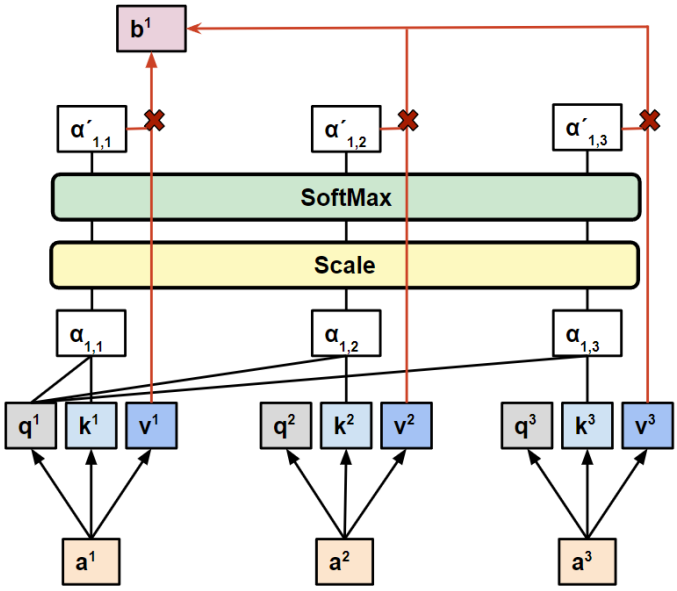


Figure 2.6: Scaled Dot-Product Attention Mechanism[44]
a is an input vector or an output vector of a hidden layer. q, k, v stand for query, key, value, respectively. The attention layer generates the output vector b, which is a weighted representation that aggregates information from the entire input sequence.

Vaswani et al.[81] introduced multi-head attention in their work. Each head aims to capture a certain attention pattern for a better understanding of the complex relationships[82][25]. Figure 2.7 explains the calculation process of the multi-head attention. In Figure 2.8[65], it is clearly proven that each token in different heads is related to other tokens in different ways, highlighting the ability of the multi-head attention mechanism to capture various relation patterns.

2.2.2 *Transformer Adapted for Computer Vision*

The widespread application of transformers in a variety of natural language processing tasks inspired the development of Vision Transformers (ViTs), which have in recent years demonstrated potential performance in computer vision tasks, such as segmentation, object identification, and recognition[72]. The ViTs were first introduced in the pioneering research of Dosovitskiy et al.[33], in which the transformer architecture is directly applied in the field of computer vision with minimal possible adaptions.

Figure 2.9 illustrates the modified model architecture. The transformer was originally introduced for handling sequence token embeddings and has been adapted by Dosovitskiy et al. to process images effectively. The most important adaption involves image tokenization, where an image $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$ is considered being equivalent to a sequence containing N patches $x_p \in \mathbb{R}^{N\times\left(P^2\cdot C\right)}$. The number of patches $N = HW/P^2$, (H, W) represents the reso-

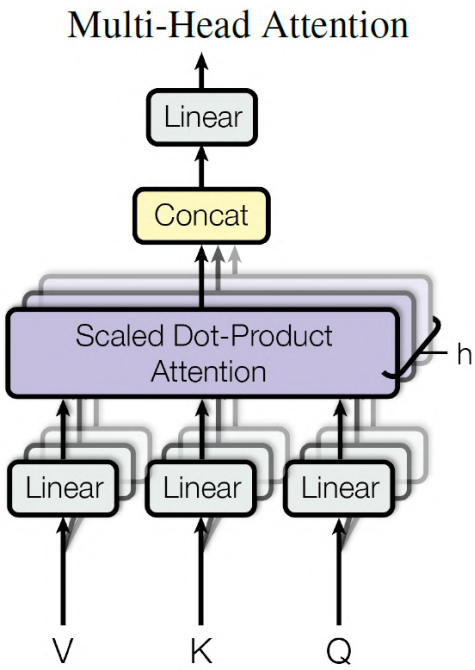


Figure 2.7: Multi-Head Attention[81]

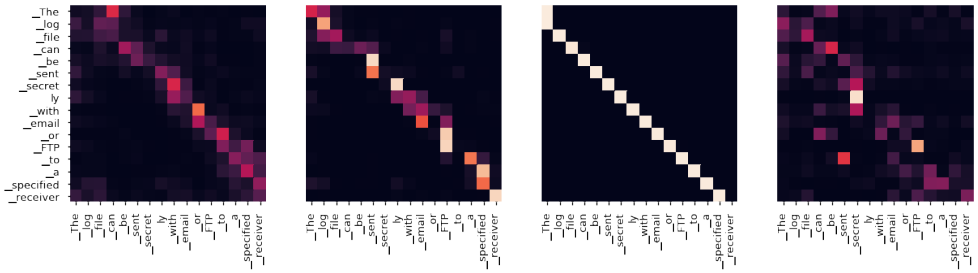


Figure 2.8: Visualization of the Attention of Different Heads at the Same Layer[65]

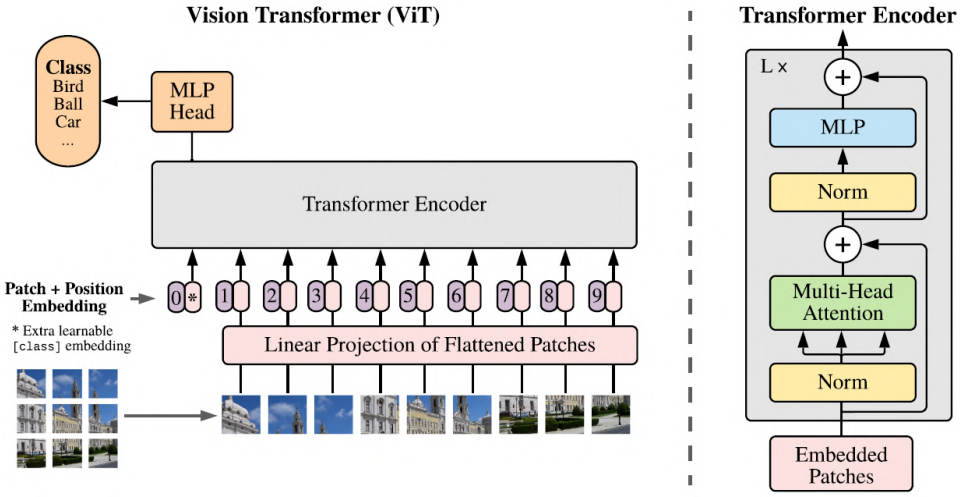


Figure 2.9: Vision Transformer Model Overview[33]

lution of the original image and (P, P) denotes the resolution of each patch. After being flattened, these patches are projected linearly to patch embeddings of dimension D through a fully connected layer. The obtained patch embeddings, together with class and positional embeddings also of dimension D, serve as input for the transformer encoder.

Instead of using the sinusoidal positional embeddings employed in the original Transformer released by Vaswani et al.[81], the authors opted to train learnable positional embeddings in the ViT, which was inspired by BERT[32]. Furthermore, as the ViT in this research is for classifying objects, the authors also added a class embedding[32]. Although the authors proposed ViT in their research initially for classification tasks, they also presented the possibility to further explore the model for dealing with other computer vision tasks, such as object detection and segmentation.

2.2.3 *Application of Transformer in VOS*

Semi-supervised VOS aims to predict segmentation masks for the objects of interest in a video sequence. The ground-truth masks for the objects of interest are provided in the first frame and used as initialized input for the following tracking[97]. The underlying concept of VOS is to acquire the relationships between the current frame (query frame) and the historical memorized frames (reference frames)[60]. Motivated by the great success of transformer in the computer vision field, researchers have started to apply it for VOS tasks[93]. With the built-in attention mechanism of transformer, a matching map $Attention(Q, K, V)$ can be calculated between the extracted features from the present frame and the memorized representations from the historical frames, where Q represents the query embedding of the present query frame, K and V stand respectively for the key embedding and the value embedding of the historical memorized frame[90][60][21].

Mei et al.[60] proposed in their work a transformer-based VOS model "TransVOS", achieving a $J\&F$ score of over 75%. Acknowledging that historical information has to be saved and might cost extensive memory, Cheng and Schwing[21] published the VOS model "XMEM". In contrast to the previous transformer-based research where the high-resolution features were merged into the feature memory after they were merged, Cheng and Schwing[21] suggest a memory system containing three independent components, namely a sensory memory updated every frame, a high-resolution working memory updated every *nth* frame, and a highly compact long-term memory that stores consolidated information from the high-resolution working memory when the latter reaches its memory limit. With the innovative memory system, "XMEM" addressed the issue of memory consumption when tracking targets in long-term videos[21]. "XMEM" reached a $J\&F$ score of over 80% on their test videos.

2.2.4 *Comparison Between CNNs and Transformer*

When compared to CNNs, which have de facto dominated the field of computer vision since 2012[46][49], ViT has its own advantages and disadvantages. CNNs benefit from specific inductive biases, such as locality and translation equivariance, which allow them to perform effectively even with limited quantities of data[26]. Differing from CNNs, ViT is built with less image-specific inductive biases. While this indicates a reduced demand for image-specific domain knowledge, it also requires a larger volume of data for effective learning to compensate for the absence of these inductive biases[33]. As illustrated in Figure 2.10, by limited dataset size, CNN exhibits a better performance compared to ViT. ViT outperforms CNN if both are trained with a very large dataset.

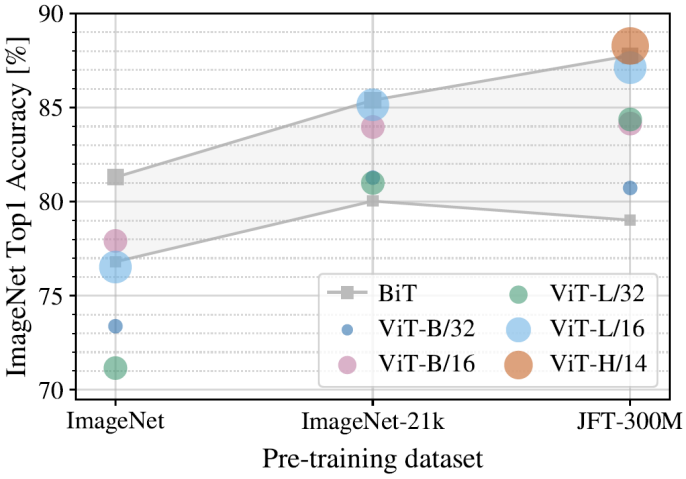


Figure 2.10: Performance of Transformer and ResNet(BiT) on Different Size of Data[33]

Another significant difference between Transformers and CNNs lies in their receptive fields in low layers. CNNs, especially in their lower layers, are only

able to capture local information among neighboring pixels since the receptive field is strongly restricted by the convolutional kernel[52]. In contrast, Transformers, due to their built-in self-attention mechanism that captures information from the entire input, are able to aggregate global information from the whole input sequence even in their lower layers. Figure 2.11 visualizes the attended distance at different layers of ViT.

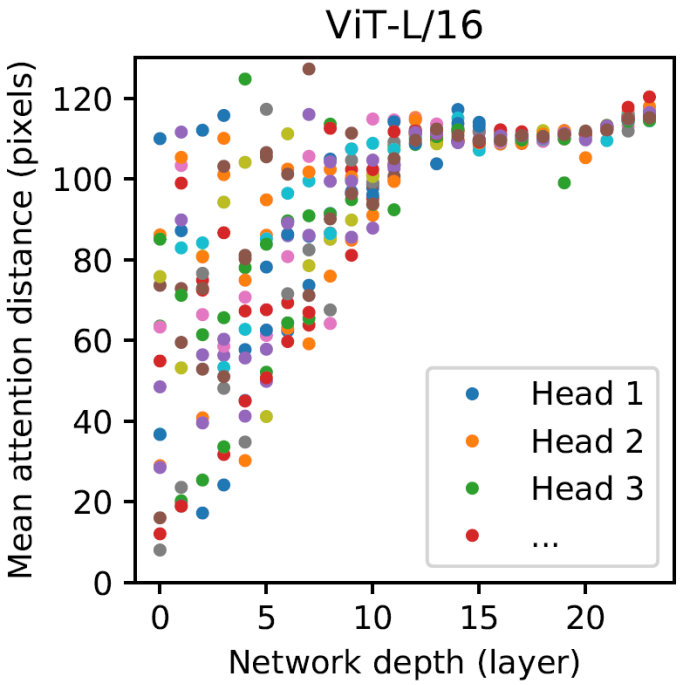


Figure 2.11: Attended Area by Different Heads and Network Depth[33]

The unsaturated performance of the ViT, as demonstrated in the research conducted by Dosovitskiy et al.[33] in which the performance of ViT continues to improve when more training data is available, indicates the potential of ViT as a foundation model in the computer vision field[9].

Moreover, the development of ViT models has greatly facilitated the interaction between the natural language processing field and the computer vision field, enabling the development of a more efficient architecture of multi-modal models[4]. In contrast to previous multimodal research that frequently relied on CNN backbones for extracting region or grid features, Kim et al.[50] introduced "ViLT", a new vision and language model based on ViT, in which images can now be linearly projected to patch embeddings that can be seen as extracted features[68], along with text embeddings as input for a unified transformer model for learning the relationship between embeddings of different modalities.

2.3 RELATED WORKS

Numerous studies regarding object segmentation and tracking have been conducted in various domains such as city traffic monitoring, self-driving

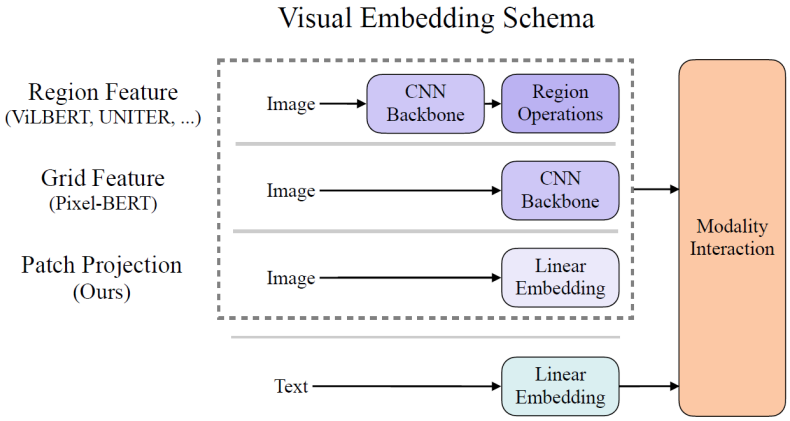


Figure 2.12: Multi-modal Models[50]

vehicles, or pedestrians tracking for public security [30] [94] [61] [35]. Many of these studies have employed frameworks built upon Mask R-CNN [39], an instance segmentation model that extends the two-stage structure of Faster R-CNN. In addition to the existing branch for bounding box generation, Mask R-CNN introduces a third parallel branch to predict binary object masks for each Region of Interest (RoI)[39].

Since its publication, Mask R-CNN has gained popularity in the field of object segmentation due to its robust performance[3]. Mask R-CNN has also caught the attention of biologists because of its effectiveness. Kassim et al.[47] conducted an investigation to explore the potential of Mask R-CNN for wild turkey detection with UAV (unmanned aerial vehicles). Despite challenging factors such as the high habitat complexity and the small target object size, Mask R-CNN achieves a F1 score of over 80%. The research of Xu et al.[47], in which Mask R-CNN was applied for monitoring livestock, demonstrated a promising performance with an accuracy exceeding 90%.

Besides two-stage models like Mask R-CNN, one-stage segmentation models, also known as end-to-end segmentation models, are an alternative for object segmentation. By omitting bounding box detection and feature repooling, one-stage segmentation models aim to simplify the traditional two-stage instance segmentation process[48]. YOLACT (You Only Look At CoefficienTs), a real-time one stage segmentation model, introduces the concept of prototype masks and per-instance mask coefficients to generate the final instance masks by linearly combining these prototypes with the corresponding coefficients[8]. SOLO (Segmenting Objects by Locations) is another state-of-the-art stage instance segmentation model that functions based on the quantized center locations and object sizes, assuming that different instances within an image are usually located in different places or have different object sizes[85][48].

These one-stage segmentation models have demonstrated their effectiveness in real-time wildlife detection applications. Choudhury et al.[24] compared the performance of YOLACT, YOLACT++, and Mask-RCNN in detecting rhinos. In their use case, YOLACT outperformed Mask-RCNN both in

speed and accuracy. Further evidence for the viability of one-stage segmentation models in real-time wildlife detection and segmentation is provided by the experimental study of Bello et al.[7]. They analyzed the performance of Mask-RCNN and SOLO in their study, finding that SOLO-based modified frameworks performed slightly better.

However, research on segmenting and tracking wildlife is still very limited due to a lack of ground-truth datasets for wildlife segmentation[73]. The lack of wildlife segmentation datasets poses a great challenge when training the aforementioned models.

Kirillov et al.[51] released SAM, a foundation model for segmentation tasks, offering a promising solution to solve the limited availability of wildlife datasets and opening up new possibilities for wildlife segmentation.

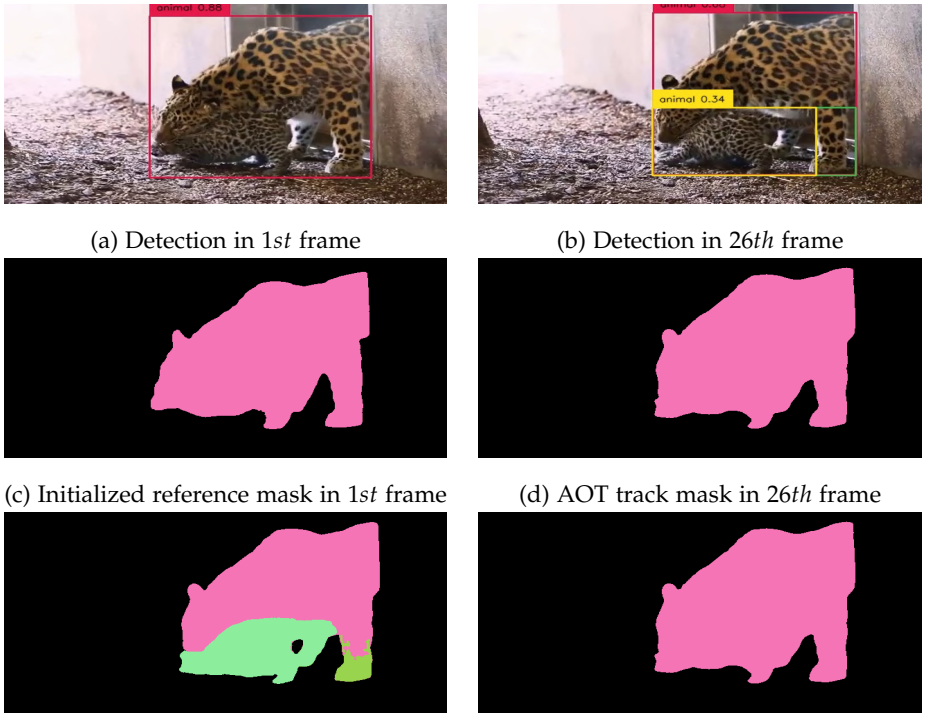
SAM consists of a flexible prompt encoder that can handle various kinds of prompts, an image encoder, and a fast mask decoder. SAM employs the ViT pre-trained with Masked AutoEncoders (MAE)[38] as its image encoder backbone. Its decoder is based on a bidirectional cross-attention mechanism that calculates cross-attention from tokens to image embedding as well as from image embedding to tokens in order to achieve a better understanding of cross-modality feature representations[51][29][86]. With only two layers, the lightweight structure allows for a very fast mask prediction[51]. Another highlight of this research is the release of the largest segmentation dataset, "SA-1B", comprising more than 1 billion masks for 11 million images that were used to train the model. Based on the training with this to date largest segmentation dataset, SAM has proven to provide strong zero-shot[88] generalization on previously unseen object classes[51].

Furthermore, SAM adopts various types of prompts, including points, boxes, masks, and text. Along with its impressive zero-shot generalization ability, this enables it to act as a foundational model that can be seamlessly applied to a wide range of downstream tasks, such as instance segmentation and Text-to-Mask[51]. Yang et al.[89] published in their research Track Anything Model (TAM), an interactive VOT model that takes the SAM generated masks as initialized reference masks and tracks the annotated target objects in the following frames using XMEM, which is a semi-supervised VOS model[21]. TAM allows users to easily initialize masks with clicks and make mask adjustments interactively during the tracking process.

TAM inspired the development of other VOT models, such as SAMTrack[22] and TDeva[20] . In comparison to TAM, SAMTrack uses in its pipeline the model "DeAOT" [90] as its tracking model, which has been demonstrated to be more efficient than XMEM [92]. Furthermore, both SAMTrack and TDeva serve as multimodal tracking models allowing for text-prompted object tracking and segmentation through the combination of Grounding Dino[53], SAM, and a VOS model. The concept is that Grounding Dino initially generates detection boxes, which are then used in SAM as prompts for generating initialized segment masks. These masks are employed in a semi-supervised VOS model to track objects in the following frames. To guarantee the detection and tracking of newly appearing objects, the masks are regularly updated.

According to SAMTrack, a new object is defined as an object that appears in the background for the first time. This means that although SAMTrack might be effective when objects do not overlap significantly, it may miss a new object if this new object heavily overlaps with an already existing object. Figure 2.13 (a) to (f) illustrate a failed example of SAMTrack. In the initial frame, two leopards are detected as a single animal due to the overlap (a), resulting in the generation of only one leopard mask as the initialized mask (c), which is then used for tracking in the following frames. In the 26*th* frame, where an update happens, although two leopards are detected (b) and SAM also segments two leopard masks (e), SAMTrack still only recognizes one leopard. Ideally, one would expect that the AOT Track mask (d) of the 26*th* frame could be modified by the SAM mask and thus two leopards can be segmented and tracked in the next frames. However, the limitation of SAMTrack lies in its definition of new objects. As a result, the overlapped smaller leopard is not considered as a new object.

TDeva addressed the issue of overlapping to some extent by identifying a new object as one having a low IoU(<0.5) with the previous segmented objects. Nevertheless, as shown in Figure 2.14, the overlap in the initialized frame still results in missegmentation until a re-segmentation is performed.



(a) Detection in 1*st* frame

(b) Detection in 26*th* frame

(c) Initialized reference mask in 1*st* frame

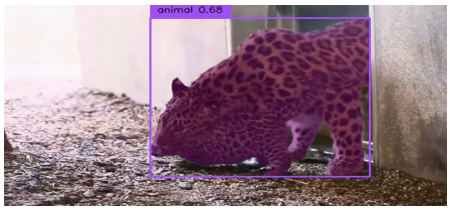
(d) AOT track mask in 26*th* frame

(e) SAM segmentation mask in 26*th* frame

(f) Updated reference mask in 26*th* frame

Figure 2.13: A Failed Example of SAMTrack

Both SAMTrack and TDeva are multimodal tracking models with Grounding Dino as detector. Grounding Dino has a transformer-based structure and was pre-trained as a zero-shot detection model, enabling it to track "anything" using text prompts. On the other hand, Grounding Dino may



(a) Segmentation mask in 1*st* frame

(b) Segmentation mask in 23*th* frame

(c) Segmentation mask in 26*th* frame

Figure 2.14: A Failed Example of Tracking with DEVA

generate a single bounding box for multiple objects due to the nature of transformer (see Figure 2.15 and Figure 2.16). The multiple animals detected within a single bounding box are segmented and treated as a single animal since they are assigned the same mask value, which not only increases the likelihood of mis-classifying an already existing object as a newly appearing object but also poses a challenge in identifying individual wildlife.

Figure 2.15: Segmentation Example Frame with SAMTrack (Grounding Dino as Detector)

2.4 GOAL OF THIS THESIS

This thesis aims to:

1. Present a general VOT framework called "MegaCutie", which combines "MegaDetector"[4] , "SAM", and "Cutie"[19] to automatically segment and track wildlife in videos, supporting biologists in further biodiversity surveys such as wildlife identification.

Figure 2.16: Segmentation Example Frame with TDeva (Grounding Dino as Detector)

2. Address the overlapping issues during tracking with a matching algorithm.

3. Propose a post-processing algorithm to address the misdetection or overlapping issue at the first frame.

3

DATA AND FRAMEWORK

In this section, the test data and the proposed framework are described in detail.

3.1 DATA SAMPLE

Experiments in this thesis were carried out with different videos, which can be mainly classified into two types:

1. High-resolution Videos (1920x1080) containing only a single animal

   The test videos, a collection of high-resolution leopard videos with static background, were provided by the "Pan African Programme"[71]. Challenging videos are selected to test the robustness of the proposed framework. The target animals are leopards in complex environments. Two of these videos were filmed under very extreme illumination conditions. The duration of each video is approximately 10 seconds. Figure 3.1 shows some test samples.

(a) Leopard in dark

(b) Leopard in bright sunlight

(c) Leopard in wood

(d) Leopard in wood

Figure 3.1: Leopard Test Samples

2. Low-resolution Videos (480x360) containing multiple animals

   The real-world environment is usually complex and the video quality might also be limited. Since the framework is developed for general tracking tasks for wildlife research, two wildlife test videos of relatively low resolution were selected from YouTube to evaluate the model. In these test videos, multiple animals are heavily overlapped. The test samples are shown below in Figure 3.2.

(a) Overlapping leopards

(b) Overlapping wildlife

Figure 3.2: Overlapping Wildlife Test Samples

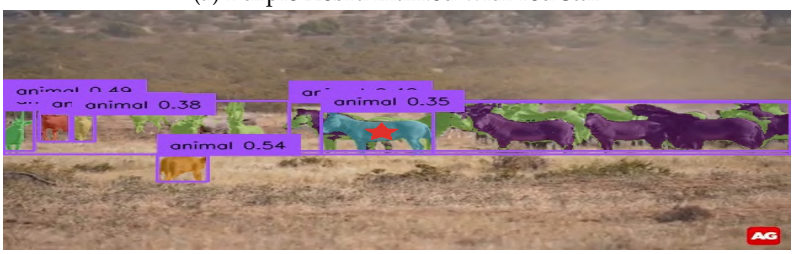
3.2 FRAMEWORK

3.2.1 *Applied Models in the Framework*

Similar to the fundamental concept of previous works[22][20], the proposed framework in this thesis consists of three components: a detection model, a segmentation model, and a tracking model. The detection model generates bounding boxes that are used as box prompts for segmentation in the segmentation model. The segmentation model then produces object masks, which are used as initialized mask in the tracking model for the further tracking.

MegaDetector is applied here as detector instead of Grounding Dino. As previously mentioned, although Grounding Dino achieves impressive performance in object detection[53], the bounding boxes provided by Grounding Dino might include multiple animals, and thus raise the risk of different animals in a bounding box being assigned the same mask value in the segmentation phase. This not only increases the risk of segmenting an already existing object as a new one, as illustrated in Figure 3.3, but also leads to difficult situations when identifying individual wildlife.

(a) Purple Zebra marked with red star



(b) Blue Zebra marked with red star

Figure 3.3: Same Zebra but Identified as Different Zebra

MegaDetector is an object detection model released by Microsoft[4] as part of the "AI for Earth program"[63]. It has been developed for finding camera trap images containing animals, people, and vehicles so that blank images can be effectively excluded, saving conservation biologists a lot of time and efforts in preparatory work[4][31]. It has been trained on huge datasets, including large private datasets as well as 13 public datasets, such as Caltech Camera Traps[13], COCO[12], and iNaturalist Dataset 2017[27][64]. The large scale and extensive diversity of training data makes MegaDetector a common tool for wildlife detection, unlike previous animal detection models that were limited to specific species.

In prior research[36], MegaDetector has also demonstrated reliable and efficient performance. YOLOV5 model is used in the most recently released version of MegaDetector, saving inference time when compared to the Faster R-CNN model that has been previously used[95]. A few challenging frames were selected to evaluate the performance of MegaDetector. As shown in Figure 3.4, it performs well even when confronted with low-resolution images with overlapping objects. Since the aim of this thesis is to build a general framework for segmenting and tracking wildlife without classifying any species, MegaDetector is used here as detector.

Figure 3.4: Detection Result of MegaDetector on a Low-Resolution Image 480x360

SAM is applied as segmentor of the framework due to its zero-shot capabilities as a segmentation foundation model[51]. Nevertheless, the emphasis lies on examining the potential of SAM for wildlife segmentation tasks since wildlife, unlike other segmentation targets, often has a highly complex background with various distractors.

Cutie is employed as tracker for the framework. In comparison with previous pixel-level VOS models, such as XMEM and deAOT, which maps the query pixels to the referenced pixels retained in memory independently, Cutie works on an object-level, and therefore uses object-level queries instead of pixel-level queries. Furthermore, object-level memory and pixel-level memory are integrated in Cutie, enabling the model to capture not only pixel-level details but also object-specific features[19]. According to the research of Ho et al.[19], Cutie can effectively reduce the noise from distrac-

tors, and thus it generates more robust object masks due to the introduction of object-level query and object-level memory. On most test datasets, it outperforms XMEM, deAOT, and TDeva in terms of both accuracy and computing time. It has proven to be approximately two times faster than deAOT and four times faster than XMEM with regard to computing time[19]. Moreover, Cutie saves considerable GPU memory when tracking targets in long-term videos[19]. Figure 3.5 demonstrates a comparison of object masks separately generated by Cutie and by XMEM. As shown in this figure, the mask produced by Cutie is obviously less affected by distractors at night.

(a) Mask generated by an object-level track model Cutie

(b) Mask generated by a pixel-level track model XMEM

Figure 3.5: Comparison of Track Masks Generated by Cutie and by XMEM
The Object Mask generated by Cutie is less affected by distractors.

3.2.2 *Pipeline*

The workflow of the suggested framework is depicted in Figure 3.6.

Initially, the first frame of an input video is fed into the framework. The detector is responsible for finding any wildlife that appears in this frame and producing the bounding boxes. To mitigate the potential issue of strongly overlapping bounding boxes and thus inaccurate segmentation masks, the inclusion rates of the bounding boxes are calculated in pairs, as shown in Formula 3.2(1). Two bounding boxes are considered strongly overlapping if their inclusion rate exceeds a pre-defined threshold value of 0.9.

$$Inclusion\,rate(B_i, B_j) = \frac{B_i \cap B_j}{B_i} \quad \text{with} \quad i \neq j. \tag{3.2(1)}$$

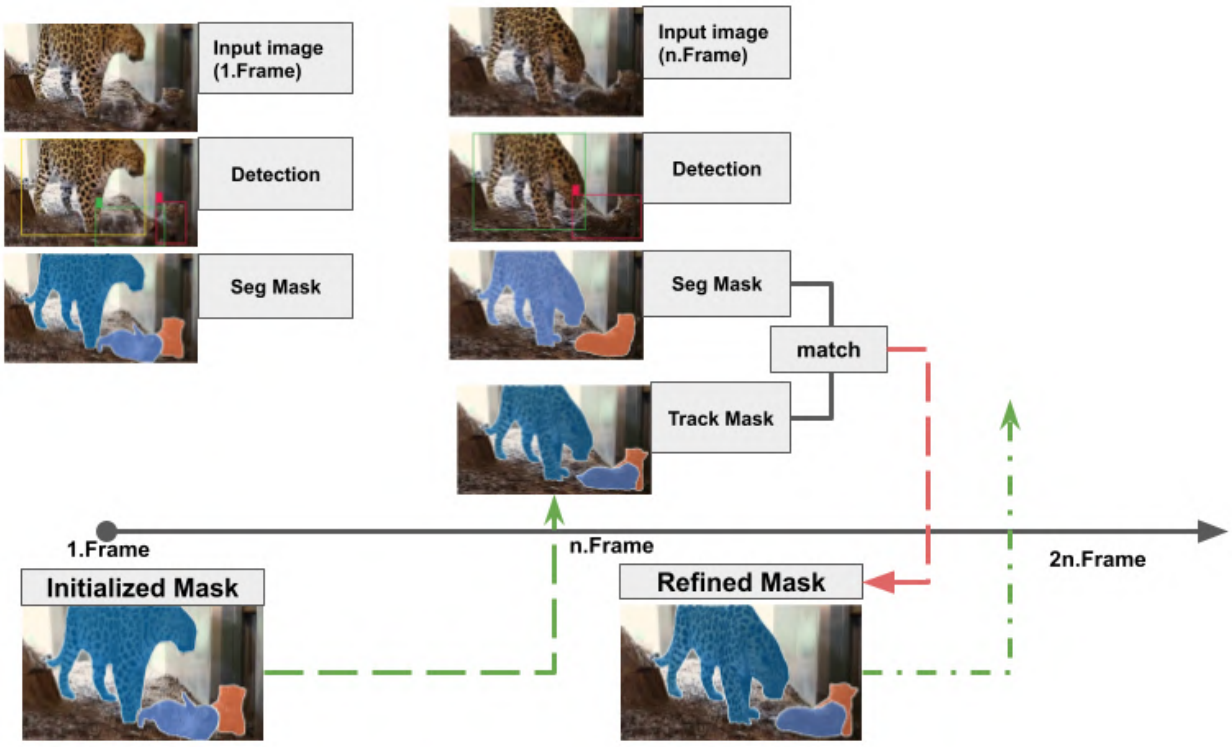
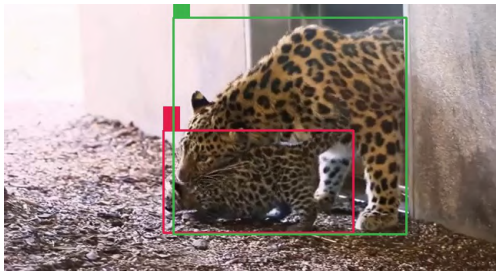


Figure 3.6: Pipeline

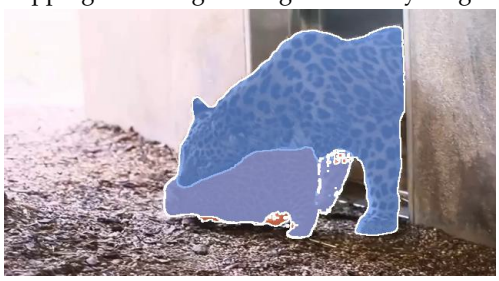
The order in which the bounding boxes are handled could influence the final segmentation result when there are overlapping bounding boxes. If the bounding box containing the larger target is segmented first, the smaller target that lies within it might be mis-segmented since the pixels that fall within the overlapping region have already been assigned to the larger object. To reduce the effect of overlapping bounding boxes, the smaller bounding box will be first used as a box prompt in SAM to generate the segmentation mask, ensuring that the smaller target can be correctly segmented. An example of segmentation with unsorted box prompts and an example for segmentation with sorted box prompts are shown in Figure 3.7.

After the segmentation masks are generated using sorted box prompts, the tracker takes these segmentation masks as initialized reference masks to follow the objects in the first frame until the nth frame, where n is a predefined update frequency. At the nth frame, the reference masks would be updated to get the refined masks, which would be used as reference masks for the next n frames. This update is required to be performed regularly for three reasons:

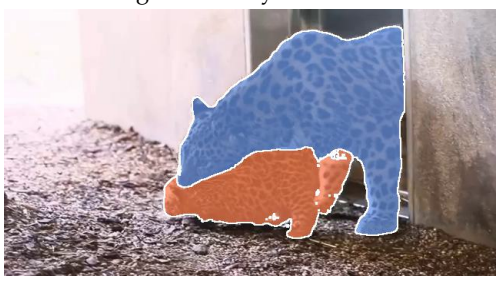
1. New wildlife might appear during tracking. Without updates, these newly appearing targets would be overlooked.



(a) Overlapping bounding boxes generated by MegaDetector



(b) Segmentation masks generated by SAM with unsorted box prompts



(c) Segmentation masks generated by SAM with sorted box prompts

Figure 3.7: Segmentation by Overlapping Bounding Boxes
In b), the small leopard is considered a part of the adult leopard since the box prompts are unsorted, the adult leopards is segmented first and SAM segments the overlapping area and the adult leopard as a single entity. In c), the small leopard is segmented first, allowing for the separation of the small leopard from the adult leopard.

2. The masks of wildlife might become imprecise during tracking and require adjustment.

3. Despite the presence of wildlife, for some reason, it might not be detected and segmented at the moment of the update.

These updated reference masks are subsequently employed for tracking in the next part of the video, from the $(n+1)$th frame to the $(2n)$th frame. More details of the update process are explained in the following subsection.

3.2.3 *Matching Algorithm*

As in subsection 3.2.2 already mentioned, the reference masks for the tracker are regularly updated. The update is based on the SAM segmentation mask, which is generated at every nth frame, and on the track mask generated by

Cutie at the same frame. Using the SAM segmentation mask and the Cutie track mask, the update produces the refined masks that subsequently serve as new reference masks for the tracking in the next n frames. More visualizations of the updates are attached to the Appendix.

Because the same object might be assigned different mask values in SAM segmentation masks and Cutie track masks, the objects are required to be mapped first. The mapping starts with the calculation of Mask Intersection-over-Union (Mask IoU). This metric is computed by dividing the intersection area of the masks by their union area[17]. The Mask IoU provides a measure of similarity between SAM segmentation and Cutie track masks for the existing objects and is mathematically represented as follows:

$$Mask\ IoU(M^{SAM}, M^{Cutie}) = \frac{|M^{SAM} \cap M^{Cutie}|}{|M^{SAM} \cup M^{Cutie}|} \tag{3.2(2)}$$

- Mapping case 1: exact One-To-One match

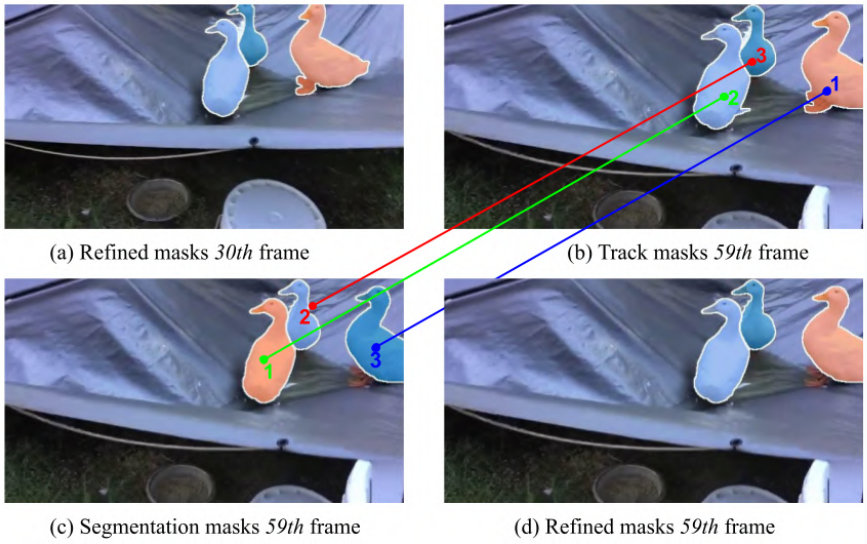
For each SAM segmentation mask, the mapping process attempts to identify a Cutie track mask that has the highest Mask IoU with the given SAM mask. The pair consisting of a segmentation mask and a track mask with the highest IoU $\geq$ 0.85 is considered an exact One-To-One match, implying that these two masks point exactly to the same object. In case of an exact One-To-One match, the refined mask for an object is determined by the SAM segmentation mask, considering that noise might appear in the track mask over time, and thus the object mask needs to be slightly modified.

- Mapping case 2: appearance of new objects

New objects might appear in a video sequence at any time. In such a scenario, the SAM segmentation mask of a new object fails to match any already existing track masks. A new object mask is defined here as the SAM segmentation mask with the highest Mask IoU less than 0.3 when compared to all currently existing Cutie track masks.

- Mapping case 3: objects presented but undetected

Objects which were detected and tracked in previous frames may become heavily overlapped due to their movement. In the update phase, the MegaDetector may fail to detect these objects, leading to missegmentation in SAM. In Figure 3.10 (b), wildlife within a red circle is incorrectly classified as part of the background in the SAM since there are no bounding box prompts generated by the MegaDetector for these overlapped animals. To prevent the omission of the overlapped targets during updating, the Cutie track masks are considered to represent targets that are currently still existing but are

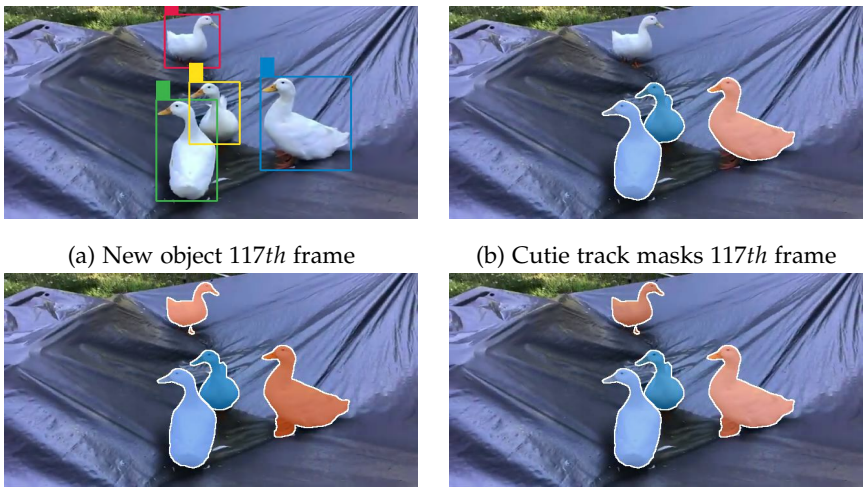


(a) Refined masks *30th* frame (b) Track masks *59th* frame

(c) Segmentation masks *59th* frame (d) Refined masks *59th* frame

Figure 3.8: An Illustration of Mapping between Segmentation Masks and Track Masks

(a) The refined masks at the 30*th* frame used for tracking in frames 31 to 59; (b) The Cutie-generated track masks, which have the same mask values as the initialized masks in (a); (c) The segmentation mask produced by SAM; (d) The refined masks after mapping, which are based on (b) and (c) for subsequent tracking.



(a) New object 117*th* frame (b) Cutie track masks 117*th* frame

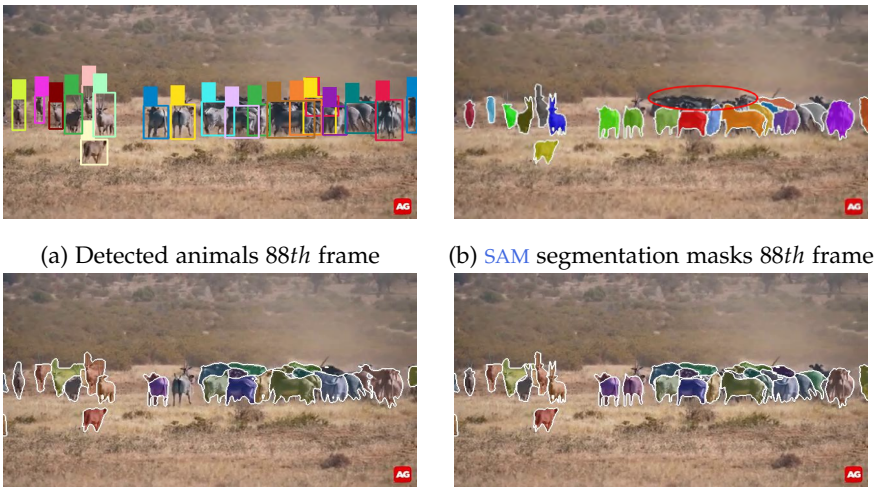
(c) Segmentation masks 117*th* frame (d) Refined masks 117*th* frame

Figure 3.9: New Animals During Tracking

(a) A recently detected duck within the red bounding box; (b) Already existing track masks in the 117*th* frame; (c) The segmentation mask generated by SAM for the 117*th* frame: the newly detected duck is segmented; (d) Updated refined masks for later tracking: adding the new segmentation mask to the currently existing track masks.

undetected at the moment of the update if their highest Mask IoUs are less than 0.3 when compared to all existing segmentation masks.

- Mapping case 4: adjustment of object masks



(a) Detected animals 88*th* frame (b) SAM segmentation masks 88*th* frame

(c) Cutie track masks 88*th* frame (d) Refined masks 88*th* frame

Figure 3.10: Mis-Detected Objects

(a) MegaDetector detected animals within bounding boxes; (b) SAM segmented masks in the 88*th* frame. Animals within the red circle are not segmented as they were not detected by the MegaDetector in (a); (c) Cutie track masks in the 88*th* frame. The animals that were considered a part of the background by SAM in (b) are still tracked by Cutie in (c) since they were detected and segmented in the previous frame; (d) Updated refined masks for the following tracking: Cutie track masks are kept for the mis-segmented targets.

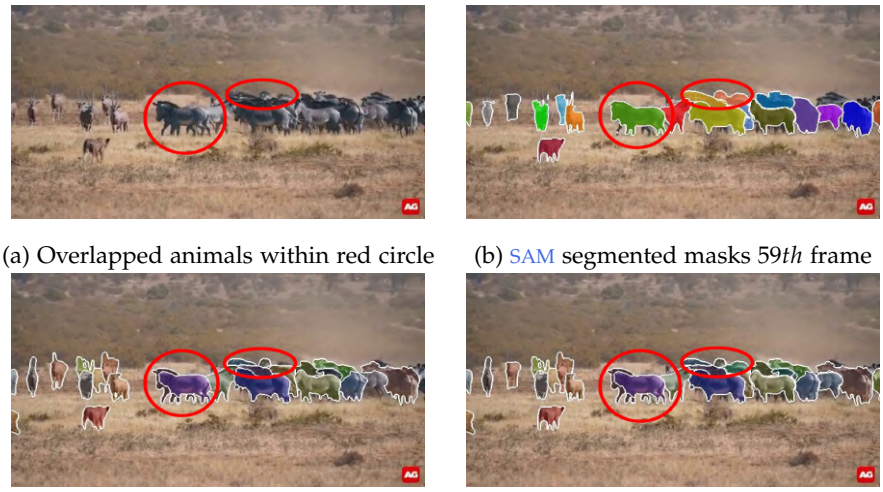
■ 4.1 Adjustment of object masks in One-To-N match

In the final situation, the Mask IoU between the SAM segmentation mask and the Cutie track mask for a given object falls within the range between 0.3 to 0.85. This often happens in a One-To-N match, which means that one given SAM segmentation mask might be mapped to multiple objects represented by different Cutie track masks, or vice versa, a single Cutie track mask could be associated with several SAM segmentation masks. Overlapping targets are a common cause of these One-To-N cases. It is noteworthy here that the Cutie track masks, which have already been handled in the previous three mapping cases, would be ignored in the last situation. Figure 3.11 demonstrates a One-To-N example.

To address the issue of overlapping, besides Mask IoU, another metric is introduced here, namely the Mask inclusion rate that is expressed below:

$$InclusionRate^{SAM} = \frac{|M^{SAM} \cap M^{Cutie}|}{|M^{SAM}|}; InclusionRate^{Cutie} = \frac{|M^{SAM} \cap M^{Cutie}|}{|M^{Cutie}|} \quad (3.2(3))$$

Compared to the metric Mask IoU, the mask inclusion rate is an asymmetric and bidirectional metric that could be more effective in handling overlapping situations. Relying solely on Mask IoU may not provide sufficient information to recognize where the overlapping happened and the direction



(a) Overlapped animals within red circle (b) SAM segmented masks 59*th* frame

(c) Cutie track masks 59*th* frame (d) Refined masks 59*th* frame

Figure 3.11: An Example of One-To-N Match

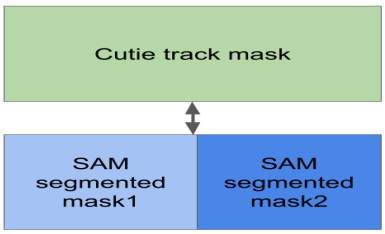
(a) Overlapped animals within red circles in the 59*th* frame; (b) The animals within the red circles share the same mask value because they were segmented as one instance by SAM; (c) Cutie tracks these animals individually since they were separately segmented by SAM in the course of the previous update; (d) Refined masks based on (b) and (c).

of the One-To-N relationship. As illustrated in Figure 3.12, both situations (a) and (b) show a Mask IoU of 0.5 between the SAM segmentation mask and the Cutie track mask. However, (a) represents a situation where overlapping might have occurred in the previous frame. One Cutie track mask is currently mapped to N SAM segmentation masks. (b) indicates a situation where overlapping might be happening in the current frame, with one SAM segmentation mask mapped to N Cutie masks. Although the Mask IoU values are identical in both situations, the Inclusion Rates differ, contributing to a more effective handling of overlapping cases.

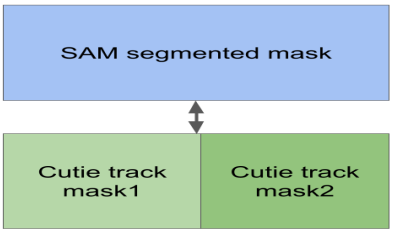
The basic concept of One-To-N Matching Algorithm is represented in pseudocode form in Algorithm 1. A step-by-step calculation example can be found in the Appendix in Figure A.1.

Since a specific SAM segmentation mask can be mapped to several Cutie track masks, j denotes all the Cutie track masks that are mapped to one given SAM segmentation mask, and the mapped Cutie track masks are sorted in descending order based on their Mask IoUs with the given SAM segmentation mask. This means the **first mapped Cutie mask** always has the highest Mask IoU with the given SAM Mask. For each $SAM\ Mask_i$ exists one $Cutie\ Mask_{h_i}$ that has the highest Mask IoU with the $SAM\ Mask_i$. The SAM Masks are also sorted in descending order based on their Mask IoUs with $Cutie\ Mask_{h_i}$.

For a given SAM Mask, the sum of $InclusionRate_j^{\text{SAM}}$ is computed. If this sum falls below a predefined threshold of 0.7, this indicates that more than 30% of the area of the given SAM segmentation mask is not covered by the Cutie track masks. As previously mentioned, 30% is also the threshold for defining a new object. In addition to the SAM segmented mask being added as a new object, the first mapped Cutie track mask will be directly kept as



(a) One track mask in Cutie maps to N segmentation masks in SAM



(b) One segmentation mask in SAM maps to N track masks in Cutie

Figure 3.12: One-To-N Match

an object mask. Other mapped Cutie masks that have not been processed or are not mapped to other SAM masks as their first Cutie masks will also be kept as object masks.

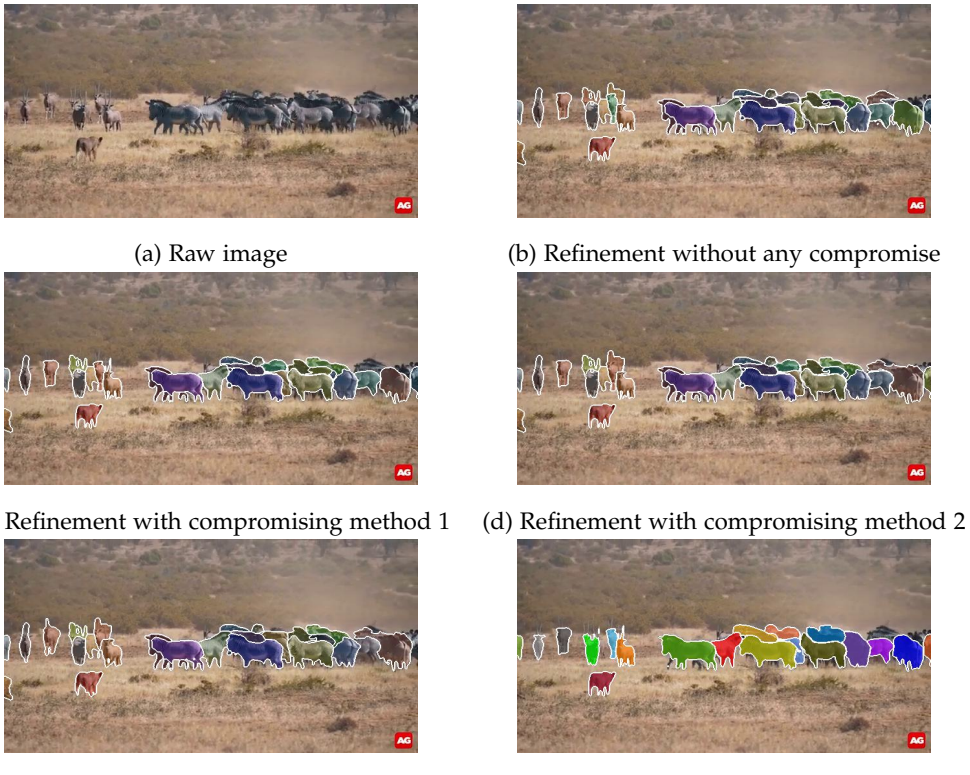
When $InclusionRate_j^{\text{SAM}}$ adds up to more than the predefined threshold of 0.7, this indicates that the mapped Cutie track masks roughly match or even fully cover the given SAM segmentation mask. Since a Cutie track mask might be mapped to more than one SAM segmentation mask, the sum of $InclusionRate_i^{\text{Cutie}}$ (in algorithm: $\sum(dictInclusion\{j\}.values)$) is computed for the Cutie track mask j. i stands for the SAM segmentation masks that are mapped to the Cutie track mask j. If the sum result is less than the threshold of 0.7, the Cutie mask j is not entirely covered by the mapped SAM masks. As previously mentioned in mapping case 3, 30% is also the threshold for defining a mis-detected object. In this scenario, besides adding the SAM masks as new object masks, the Cutie mask is kept to ensure that no object is overlooked during the update process. If the sum result is greater than or equal to the threshold of 0.7, while ignoring the corresponding Cutie track mask j itself, the first mapped SAM mask, which has the highest Mask IoU with Cutie track mask j, is added as mask modification to the already existing object, and other mapped SAM masks are added as new object masks.

Note that the Cutie masks, except the ones that are mapped for different SAM masks as their first Cutie mask, will not be repeatedly processed. Two lists are given in the algorithm, i.e., *listCutieValues* and *listCutieValues*. *listCutieValues* saves the mask values of all the $Cutie\ Mask_{h_i}$. The mask values of those Cutie track masks which have already been kept in the final refined masks are saved in *processedCutieValues*.

As previously mentioned, all the Cutie track masks that are mapped to one given SAM segmentation mask are sorted in descending order based

on their Mask IoU with the given SAM segmentation mask. The first Cutie mask has the highest Mask IoU with the given SAM segmentation mask. For a given SAM segmentation mask, its first mapped Cutie mask is directly processed. Its other mapped Cutie track masks will be checked first before they are processed. Those whose mask values that are already existing in the aforementioned two lists are to be ignored because they have been processed or will be processed, while the remaining ones are kept as object masks so that the existing objects are not mis-segmented.

Although SAM and Cutie already outperform other models, their performance still relies on the video quality. It should be noted that under the condition that $InclusionRate_j^{SAM} >= 0.7$ and $InclusionRate_i^{Cutie} < 0.7$, one object might seem to be torn to pieces if much noise appears in the Cutie mask during tracking or the generated SAM mask is inaccurate. To solve this problem, compromises could be made in two manners that are explained below. A comparison of matching results is shown in Figure 3.13. More comparisons are to be found in the Appendix.



(a) Raw image

(b) Refinement without any compromise

(c) Refinement with compromising method 1

(d) Refinement with compromising method 2

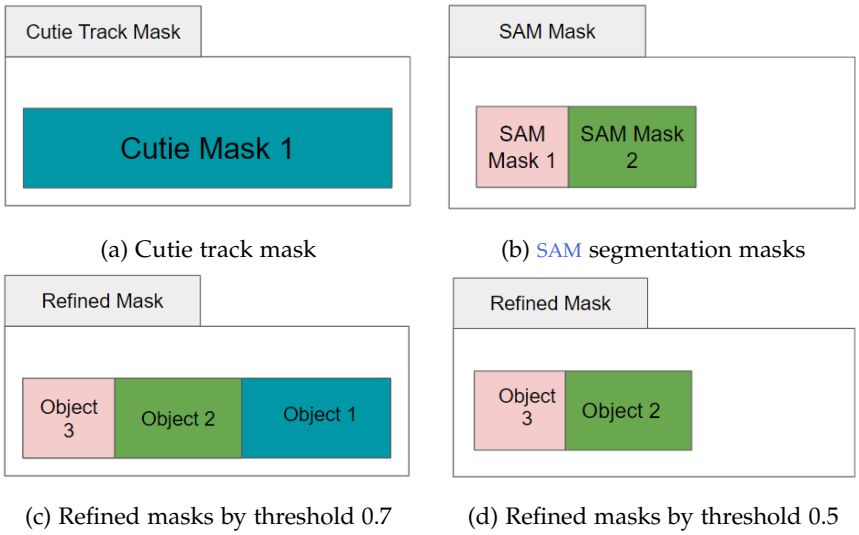
(e) Cutie track masks

(f) SAM Segmentation masks

Figure 3.13: Comparison of the Matching Results

The first method, as illustrated in Figure 3.14, is to lower the threshold value, which has the default value 0.7. By reducing the threshold, an object is less likely to be segmented in pieces, but the risk of missing objects presented by the Cutie track mask increases.

The second compromising method (see Figure 3.15) is that after keeping the Cutie mask as an object mask, all the mapped SAM masks are still added



(a) Cutie track mask

(b) SAM segmentation masks

(c) Refined masks by threshold 0.7

(d) Refined masks by threshold 0.5

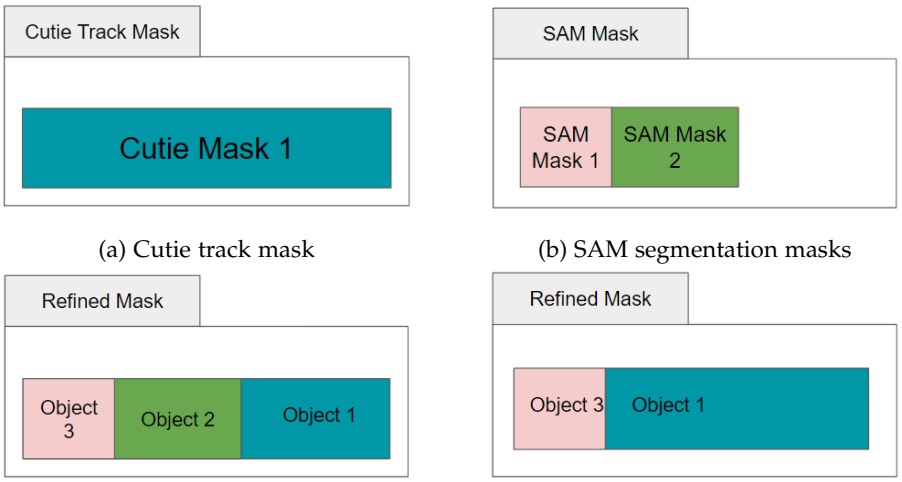
Figure 3.14: An Illustration of Compromising Method 1
Situation 1: If Cutie mask 1 contains noise, then the mask of object 1 in (c) might be noise; however, it is segmented as an object. Situation 2: If Cutie mask 1 is accurate and SAM mask 2 is smaller than the actual existing object, this leads to one object being broken into two parts. Here, object 2 and object 1 in (c) should be the same object. By setting the threshold down to 0.5, the problems described in situation 1 and situation 2 could be solved, with the risk of missing object 1.

as object masks. The first mapped SAM mask, which has the highest Mask IoU with the Cutie mask, is then assigned the same value as the Cutie mask. Compared to other mapped SAM masks, this SAM mask is most likely to be the same object that is presented by the Cutie track mask. This method could reduce the possibility that one object is broken into pieces, but at the cost of possible segmentation failure. The object which is presented by the first mapped SAM segmentation mask is not separately segmented in the final refined masks.

Finally, the object masks are sorted according to their area in ascending order. The smaller object masks are added first to the final refined masks to reduce the possibility that they would be overlapped by the larger objects.

■ 4.2 Adjustment of object masks in One-To-One match

Another scenario in which the Mask IoU lies between 0.3 to 0.85 may also occur when one SAM mask is only mapped to one Cutie mask and vice versa. Compared to the exact One-To-One match in mapping case 1 where the refined mask is determined by the SAM segmentation mask, the inclusion rate gap is employed here to determine the adjustment of object masks. Since Cutie works at the object level and significant form changes of the Cutie track masks in a short time interval are very unlikely, if the absolute gap between $InclusionRate^{SAM}$ and $InclusionRate^{Cutie}$ is greater than the predefined threshold, this indicates that the generated segmentation mask during the updating process might be inappropriate due to improper bounding



(a) Cutie track mask (b) SAM segmentation masks

(c) Refined masks without compromise (d) Refined masks with compromise method 2

Figure 3.15: An Illustration of Compromising Method 2
Situation 1: If Cutie mask 1 contains noise, then the mask of object 1 in (c) might be noise, however, it is segmented as an object. Situation 2: If Cutie mask is accurate and SAM ,ask 2 is smaller than the actual existing object, this causes one object to be broken into two parts. Here, object 2 and object 1 in (c) should be the same object. By the second compromising method, the first mapped SAM mask, i.e., the SAM which has the highest Mask IoU with the Cutie Mask, is assigned the same mask value with the Cutie mask. The problems in situation 1 and situation 2 could be solved. However, this might lead to object 2 being missed.

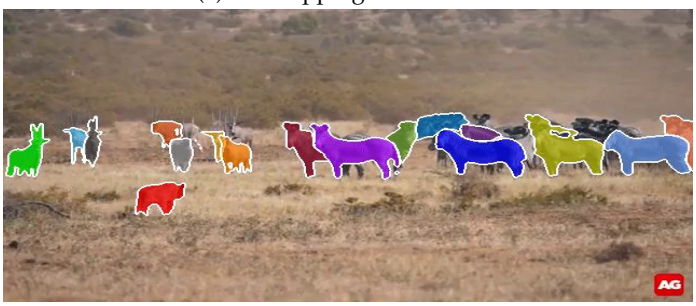
boxes or an unsatisfying segmentation quality. Despite the fact that the Cutie track mask may become slightly inaccurate during tracking, it is insufficient to consider Cutie as the cause of this relatively low match rate with the segmentation mask. In this case, the track mask will be directly used as the refined mask for the next frames. Otherwise, in case the difference between $InclusionRate^{SAM}$ and $InclusionRate^{Cutie}$ is relatively small, which is more likely due to the track mask becoming slightly inaccurate over time, the SAM segmentation mask will be used directly as the refined mask for tracking in the next part of the video. The threshold value is adjustable. A higher threshold value means more confidence in the SAM segmentation mask.

3.2.4 *Post-process*

After the first n frames, the refined masks that are formed from SAM masks and Cutie masks are employed as reference masks for tracking. However, the first part of the video (the first n frames) relies solely on the SAM-generated segmentation masks from the first frame as reference masks for tracking. Problems including misdetection and overlapping may exist, as shown in Figure 3.16. Although these issues might be addressed with future updates, the segmentation result of the first n frames still remains unsatisfying without correction.

To address the aforementioned issues, a post-process is required to improve the performance in the first part of the video. The refined masks that

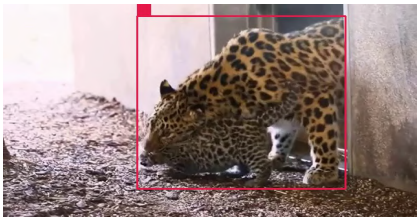
(a) Overlapping in 1*st* frame



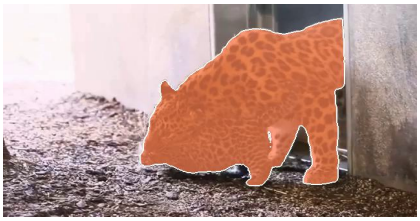
(b) Misdetection in 1*st* frame

Figure 3.16: Potential Issues in the First Frame

serve as reference masks for the second part of the video are also used as reference masks for tracking targets in the reversed frames of the first part of the video. The results after post-processing are illustrated in Figure 3.17 and Figure 3.18. As demonstrated in Figure 3.17, since the adult leopard covers the small leopard completely in the first frame, they are detected and segmented as one instance until the 26*th* frame, in which the next mask refinement occurs. The refined masks in the 26*th* frame are then used as reference masks for re-tracking targets in the reversed frames of the first part of the video so that the improper segmentation results in the first n frames can be corrected. As shown in Figure 3.17 (d), the leopards in the first frame are segmented as two separate targets after the post-processing.



(a) Leopards in one bounding box 1*st* frame

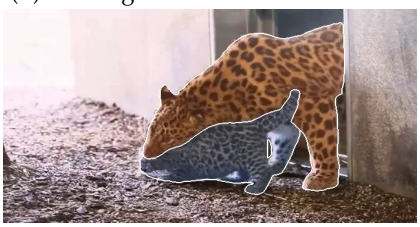


(b) SAM segmented masks 1*st* frame

(c) Cutie track masks 26*th* frame

(d) SAM segmented masks 26*th* frame

(e) Refined masks 26*th* frame



(f) Corrected masks 1*st* frame after post-processing

Figure 3.17: An Example of Post-processing

Algorithm 1 One-To-N Matching Algorithm

Requirements:

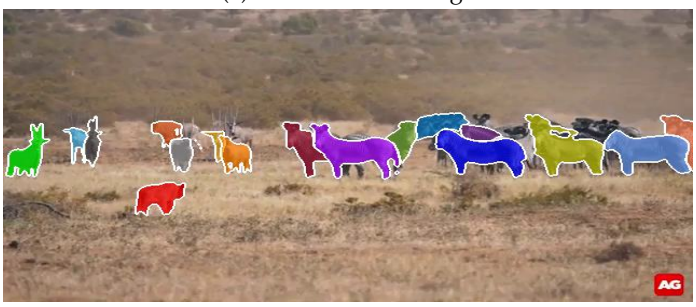
$0.3 \leq$ **Max**(**Mask IoU**($M_i^{\text{SAM}}, M_j^{\text{Cutie}}$)) < 0.85
and
$0 <$ **Mask IoU**($M_i^{\text{SAM}}, M_j^{\text{Cutie}}$) ▷ Comment: i: a mask value of SAM Mask, j: a mask value of Cutie Mask

Variables and Steps:

dictInclusionCutie = $\{j : \{i : InclusionRate_i^{Cutie}\}\ \forall i$
listCutieValues = $[h_i]\ \forall i$ ▷ Comment: for each M_i^{SAM} exists one $M_{h_i}^{\text{Cutie}}$, which has the highest Mask IoU with M_i^{SAM}, Mask IoU($M_i^{\text{SAM}}, M_{h_i}^{\text{Cutie}}$) = Max(Mask IoU($M_i^{\text{SAM}}, M_j^{\text{Cutie}}$)) $\forall j$
processedCutieValues = [ ]

for $i \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to n **do**
    if $\sum InclusionRate_j^{\text{SAM}} < 0.7$ **then**
      Add M_i^{SAM} as an object mask;
      if $j = h_i$ **then**
        Keep M_j^{Cutie} as an object mask;
      else
        if $j \notin$ *listCutieValues* and $j \notin$ *processedCutieValues* **then**
          keep M_j^{Cutie} as an object
          processedCutieValues.append(j);
        else
          pass
        end if
      end if
    else
      if $\sum(dictInclusion\{j\}.values) < 0.7$ **then**
        if $j = h_i$ **then**
          Keep M_j^{Cutie} as an object mask
          add M_i^{SAM} as an object mask;
        else
          if $j \notin$ *listCutieValues* and $j \notin$ *processedCutieValues* **then**
            keep M_j^{Cutie} as an object
            processedCutieValues.append(j);
          else
            pass
          end if
        end if
      else
        if $j = h_i$ **then**
          use M_i^{SAM} as an object mask;
        else
          if $j \notin$ *listCutieValues* and $j \notin$ *processedCutieValues* **then**
            keep M_j^{Cutie} as an object
            processedCutieValues.append(j);
          else
            pass
          end if
        end if
      end if

(a) 1*st* frame raw image



(b) Masks in 1*st* frame before post-processing

(c) Masks in 1*st* frame after post-processing

Figure 3.18: Comparison of Masks in the First Frame Before and After Post-processing

4

EXPERIMENTS AND RESULTS

4.1 QUANTITATIVE RESULTS

Five high-resolution video clips about leopards are used to evaluate the proposed framework. The performance score is calculated as the Mask IoU between the predicted mask of the framework and the corresponding ground-truth mask. Ten frames have been evaluated from each test video clip, except from the first video clip. The test results are demonstrated in Table 4.1. Even though the test videos were filmed in a complex environment with dense forests, which posed great challenges for segmentation, the framework performs still effectively and reliably. For all tested frames, it achieves a Mask IoU score of over 85%.

The median filter was applied here as a baseline method for two main reasons. Firstly, the median filter has proven to be effective for segmenting objects in a static background, which might be suitable for the given test samples since they were captured from cameras in fixed positions. Secondly, training and evaluating other segmentation models are difficult due to the lack of sufficient ground-truth datasets for wildlife segmentation. Each test video randomly took 25 selected frames from which the background is calculated as the median. To remove the small blinking noise in the background and fill the gap within the target, erosion and dilation, two common image processing techniques, were employed[67]. The results are shown in Table 4.2. However, the results are below expectations because it was not possible to estimate an accurate background using median filter in some test cases.

The statistical background subtraction models were not evaluated in this section. They are not suitable for the given test samples in this thesis since all the videos were filmed when wildlife appeared in front of the camera. For the reason that the first frame of each test sample is not free of wildlife, this would trigger the ghost area problem, as previously shown in Figure 2.3.

Furthermore, SAMTrack, as a model in previous research, is also evaluated using the same five test videos. Its quantitative results are shown below in Table 4.3. As displayed in Table 4.1 and Table 4.3, when evaluating the test frames of the high-resolution videos which contain only a single target animal, the average quantitative performance of the SAMTrack is almost on a par with the average quantitative performance of the framework proposed in this thesis. In the scenes containing obstacles, MegaCutie performs slightly better, as shown in Figure 4.1, since the tracker Cutie works on an object level.

MegaCutie	$f1$	$f2$	$f3$	$f4$	$f5$	$f6$	$f7$	$f8$	$f9$	$f10$	mean
*subject*35698457	/	0.899	0.869	0.902	0.895	0.904	0.906	0.91	0.907	0.915	0.901
*subject*35698457	0.890	0.899	0.891	0.886	0.876	0.880	0.899	0.869	0.884	0.892	0.887
*subject*35718591	0.877	0.859	0.901	0.908	0.905	0.903	0.901	0.913	0.903	0.91	0.898
*subject*35852611	0.878	0.894	0.898	0.894	0.876	0.887	0.88	0.88	0.879	0.875	0.884
*subject*35857244	0.889	0.890	0.896	0.909	0.901	0.906	0.909	0.914	0.9	0.901	0.902

Table 4.1: Performance of the Framework on the High-Resolution Leopard Videos

Baseline	$f1$	$f2$	$f3$	$f4$	$f5$	$f6$	$f7$	$f8$	$f9$	$f10$	mean
*subject*35698457	/	0.575	0.578	0.549	0.551	0.549	0.555	0.553	0.542	0.553	0.556
*subject*35698457	0.529	0.542	0.542	0.553	0.53	0.559	0.544	0.565	0.548	0.502	0.542
*subject*35718591	0.582	0.584	0.562	0.563	0.564	0.583	0.585	0.596	0.598	0.59	0.581
*subject*35852611	0.624	0.636	0.634	0.637	0.644	0.644	0.654	0.652	0.643	0.632	0.640
*subject*35857244	0.778	0.817	0.803	0.791	0.798	0.815	0.809	0.782	0.772	0.765	0.793

Table 4.2: Performance of the Median Filter on the High-Resolution Leopard Videos

SAMTrack	$f1$	$f2$	$f3$	$f4$	$f5$	$f6$	$f7$	$f8$	$f9$	$f10$	mean
*subject*35698457	/	0.844	0.847	0.858	0.845	0.853	0.856	0.864	0.857	0.861	0.854
*subject*35698457	0.890	0.903	0.886	0.873	0.876	0.874	0.894	0.868	0.879	0.890	0.883
*subject*35718591	0.878	0.859	0.894	0.896	0.897	0.897	0.896	0.905	0.892	0.893	0.891
*subject*35852611	0.908	0.907	0.909	0.908	0.889	0.901	0.89	0.897	0.89	0.886	0.899
*subject*35857244	0.887	0.897	0.911	0.917	0.911	0.915	0.914	0.919	0.905	0.914	0.909

Table 4.3: Performance of the SAMTrack on the High-Resolution Leopard Videos

(a) Raw Image (b) Result of SAMTrack

(c) Result of TDeva (d) Result of MegaCutie

Figure 4.1: Qualitative Performance Comparison in Scenarios with Obstacles at Night

4.2 QUALITATIVE RESULTS

Some qualitative results are shown in this subsection. The proposed framework is not only tested with the given high-resolution leopard test videos

but also with the YouTube low-resolution videos, which are complex scenarios with multiple overlapping animals. The qualitative results for the high-resolution leopard videos are presented in Figure 4.2 and Figure 4.3. To analyze the robustness of the proposed framework, two low-resolution wildlife videos are applied for testing.

Using the same collection of low-resolution wildlife videos, the performance of MegaCutie is compared with the performance of the models SAMTrack and TDeva in previous research. The qualitative results are demonstrated from Figure 4.4 to Figure 4.9. The displayed results of MegaCutie were generated with the second compromising method. More results that were generated without any compromising methods or with the first compromising method can be found in the Appendix.

(a) *Subject*35698457 (b) *Subject*35698457

(c) *Subject*35718591 (d) *Subject*35852611

(e) *Subject*35857244

Figure 4.2: Qualitative Results of MegaCutie on the High-Resolution Leopard Videos

4.3 FAILURE CASES

In this subsection, failure cases are analyzed, and two main reasons why failure cases occurred are presented.

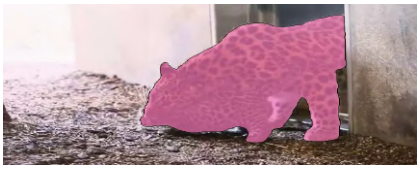
(1) Failure Case 1 : Since SAM uses detected bounding boxes as prompts for further segmentation, any misdetection could result in incorrect segmentation and tracking results. An example is shown in Figure 4.10. The missegmentation mask might disappear in a dynamic background.

(a) *Subject*32236280

(b) *Subject*35697857

Figure 4.3: Qualitative Results of MegaCutie on the High-Resolution Leopard Videos in a Challenging Environment

(a) Raw image



(b) Result of SAMTrack

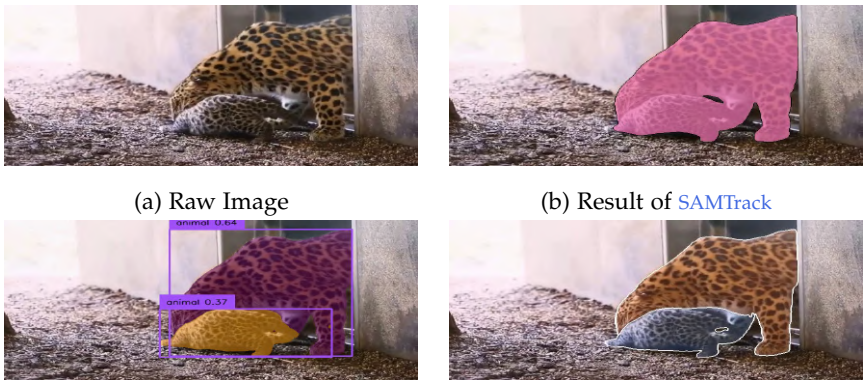
(c) Result of TDeva

(d) Result of MegaCutie

Figure 4.4: Qualitative Results of MegaCutie on the YouTube Low-Resolution Leopards Video (1*st* Frame of the Video) and Comparison With SAMTrack and TDeva

However, if the background is always static, the missegmentation mask remains as false positive in the rest frames.

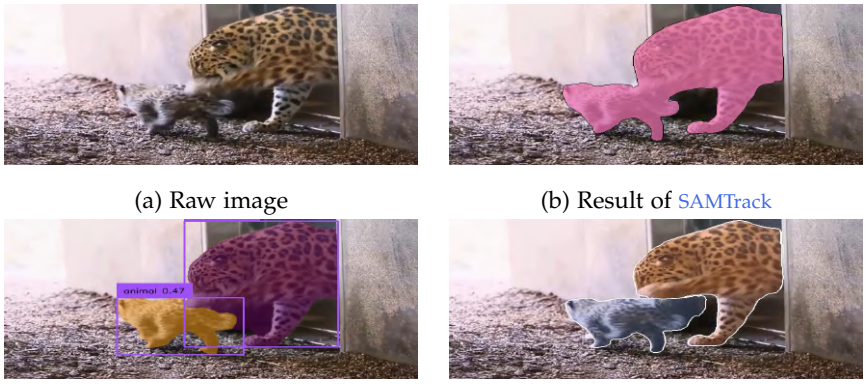
(2) Failure Case 2 : This failure occurs because the pipeline uses the pretrained SAM model, which is a foundation model for general task segmentation. Without fine-tuning for particular tasks, it may lack the specific knowledge of the expected targets. As a result, SAM might generate inaccurate segmentation masks. Two examples are shown in Figure 4.11. In Figure 4.11(b), SAM generates an over-segmentation mask when dealing with leopard spots.



(a) Raw Image (b) Result of SAMTrack

(c) Result of TDeva (d) Result of MegaCutie

Figure 4.5: Qualitative Results of MegaCutie on the YouTube Low-Resolution Leopards Video (a Frame in the Middle of the Video) and Comparison With SAMTrack and TDeva



(a) Raw image (b) Result of SAMTrack

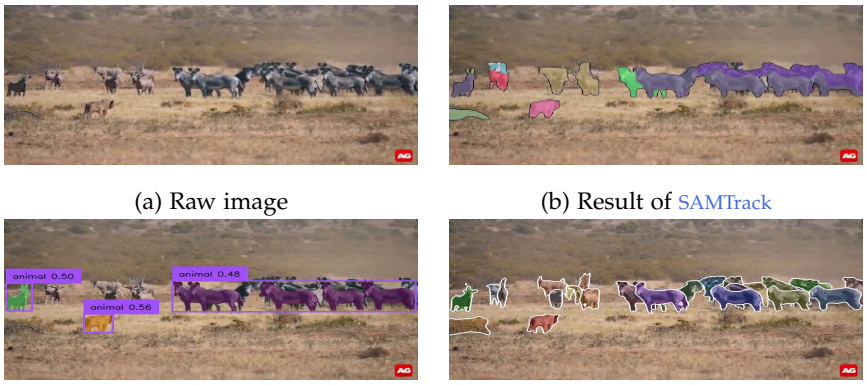
(c) Result of TDeva (d) Result of MegaCutie

Figure 4.6: Qualitative Results of MegaCutie on the YouTube Low-Resolution Leopards Video (Last Frame of the Video) and Comparison With SAMTrack and TDeva

In Figure 4.11(d), SAM generates an under-segmentation mask for the leopard at night.

4.4 APPLICATION FIELD

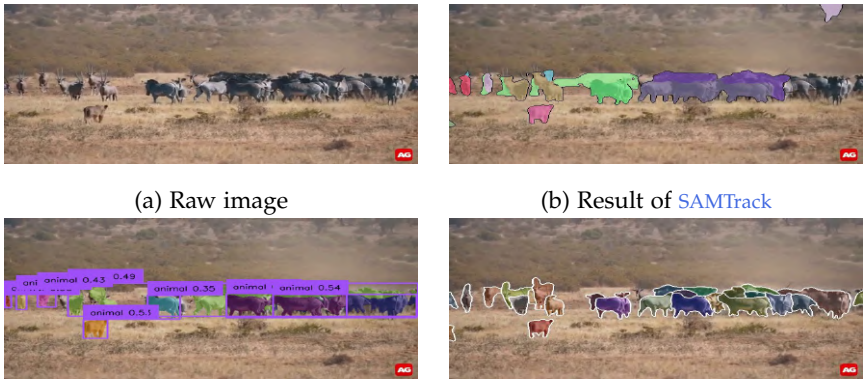
Besides automatic segmentation, the proposed framework might also be applied for automatic counting of wildlife. The counting results are shown in Table 4.4. Figure 4.12 and Figure 4.13 show the automatic counting results and manual counting results respectively. To test the performance of automatic counting, three frames were selected, i.e., the first frame of the video, a frame in the middle of the video, and a frame at the end of the video. The results show the potential of applying the proposed framework for automatic counting of wildlife.



(a) Raw image (b) Result of SAMTrack

(c) Result of TDeva (d) Result of MegaCutie

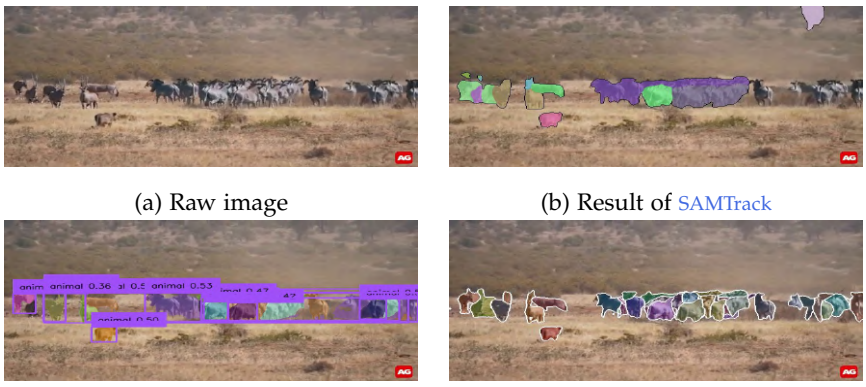
Figure 4.7: Qualitative Results of MegaCutie on the YouTube Low-Resolution Wildlife Video (1*st* Frame of the Video) and Comparison With SAMTrack and TDeva



(a) Raw image (b) Result of SAMTrack

(c) Result of TDeva (d) Result of MegaCutie

Figure 4.8: Qualitative Results of MegaCutie on the YouTube Low-Resolution Wildlife Video (a Frame in the Middle of the Video) and Comparison With SAMTrack and TDeva



(a) Raw image (b) Result of SAMTrack

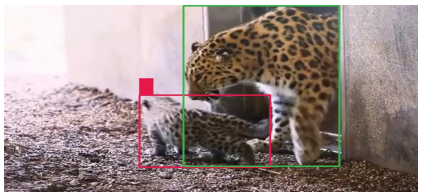
(c) Result of TDeva (d) Result of MegaCutie

Figure 4.9: Qualitative Results of MegaCutie on the YouTube Low-Resolution Wildlife Video (Last frame of the Video) and Comparison With SAMTrack and TDeva

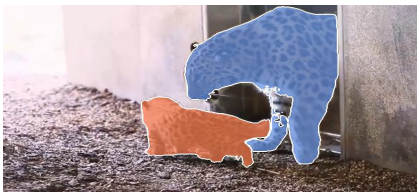
(a) Misdetection

(b) Missegmentation

Figure 4.10: Failure Case 1



(a) Leopards detection



(b) Over-segmentation

(c) Leopards detection

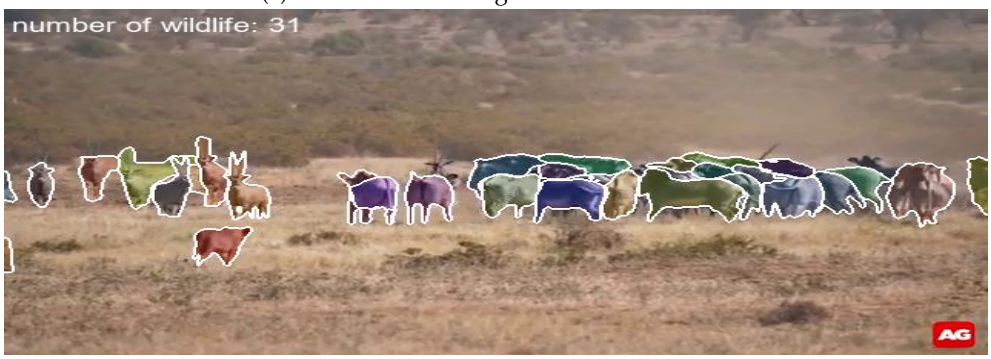
(d) Under-segmentation

Figure 4.11: Failure Case 2

Frames	**Automatic counting results**	**Manual counting results**
1*st frame*	24	26
88*th frame*	31	32
146*th frame*	35	32

Table 4.4: Automatic Counting Results vs. Manual Counting Results

(a) Automatic counting of wildlife 1*st* frame

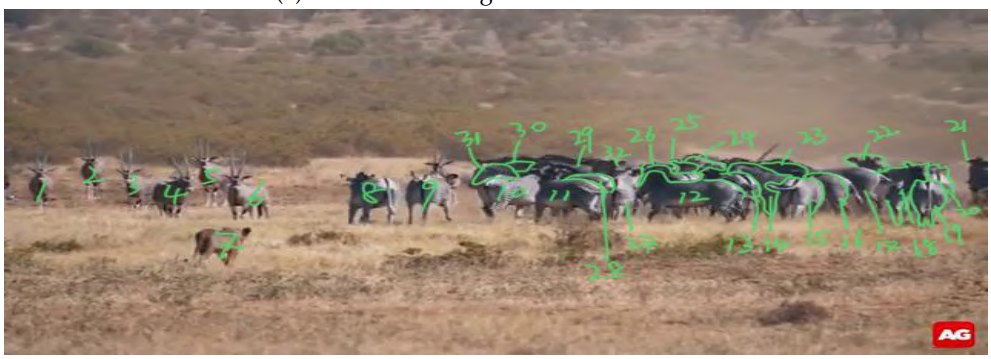


(b) Automatic counting of wildlife 88*th* frame

(c) Automatic counting of wildlife 146*th* frame

Figure 4.12: Automatic Counting of Wildlife

(a) Manual counting of wildlife 1*st* frame



(b) Manual counting of wildlife 88*th* frame

(c) Manual counting of wildlife 146*th* frame

Figure 4.13: Manual Counting of Wildlife

5 CONCLUSION

This thesis proposes MegaCutie as a framework for wildlife segmentation in video sequences. The aim is to segment and track wildlife that appears in a video sequence automatically.

The framework contains three components, namely a detector, a segmentor, and a tracker. The core concept is that the detector generates bounding boxes that are used as box prompts for the segmentor. The segmentor then generates object masks, which are used as reference masks in the tracking model for the tracking in the following frames. The reference masks are updated every nth frame to guarantee that all the currently existing objects could be segmented and tracked. To accelerate the framework, the YOLOV5-based MegaDetector is employed as detection model, while Cutie, which has proven to be more efficient than the previous VOS models both in speed and accuracy, functions as tracker. The framework integrates SAM as its segmentor because of its excellent zero-shot segmentation performance, and because of the lack of ground-truth datasets for wildlife segmentation. To address the overlapping issue, a matching procedure is conducted in every updating phase by using metrics Mask IoU and inclusion rates. Additionally, a post-process is applied in an attempt to address the overlapping issue in the first initialized frame.

To access the effectiveness and robustness, the framework is tested with high-resolution leopard videos in complex environments and with challenging illumination, as well as low-resolution YouTube videos containing multiple overlapping wildlife. The framework achieves Mask IoU scores of over 85 % with the ground-truth leopard dataset. For the low-resolution YouTube videos, it also produces reliable results in most scenarios, as demonstrated in the qualitative results in Section 3 and in the Appendix. It should be noted that there is a trade-off between fragmented object masks and possible missegmentation since both the segmentation model and the VOS model do not always work perfectly.

Nevertheless, the framework is built under the assumption that the MegaDetector generates correct bounding boxes. As the first failure case demonstrated, misdetection causes subsequent missegmentation since SAM is not trained for particular tasks and thus might lack specific knowledge for the expected target. This also leads to inaccurate segmentation in some challenging scenarios, as shown in the second failure case.

5.1 FURTHER WORK

Since wildlife often lives in a camouflaged environment and some animals are nocturnal, domain adaption[16][91] could be employed to make SAM

more robust for wildlife segmentation, and therefore the basic foundation model could be more suitable for specific downstream tasks. With domain adaption, SAM might be more reliable for segmentation of animals in complex nature surroundings or in low-light conditions.

Prompts, as the first step of the framework, are of prime importance. If performance time is ignored, different detection models could be combined so that convincing bounding boxes are more likely to be generated.

MegaCutie also shows potential for automatic counting of wildlife. More tests could be conducted in the future to test the robustness of the automatic counting function.

Part II

APPENDIX

APPENDIX

A.1 APPENDIX MODEL VERSION

Model version: Table A.1

MODEL	VERSION
MegaDetector	*md_v5a.0.0.pt*[59]
SAM	*sam_vit_h_4b8939*[62]
Cutie	*Cutie_v1.0*[18]

Table A.1: Model Version

A.2 PYTHON ENVIRONMENT AND PACKAGES

Python version: 3.9.16.
The packages have been listed in the file "requirement.txt".

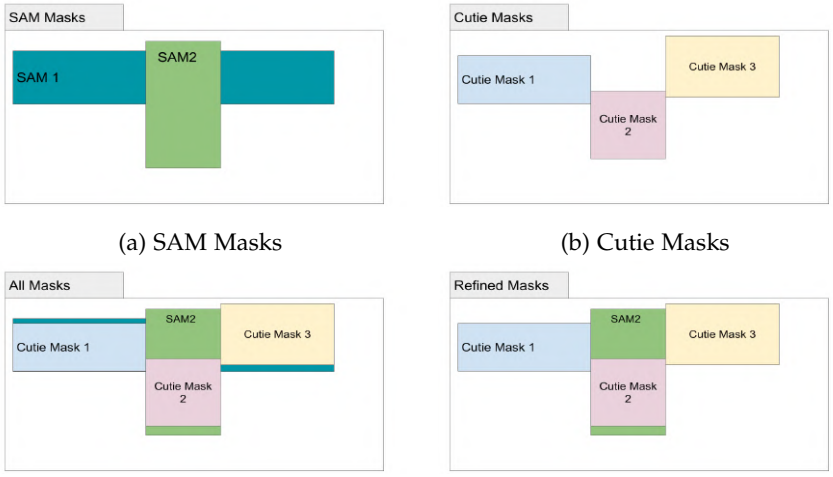
A.3 TEST VIDEOS

Links of the test videos: Table A.2

VIDEOS
Subject 35698457 (night) [103]
Subject 35718591 (daylight) [105]
Subject 35852611 (daylight) [104]
Subject 35857244 (daylight) [106]
Subject 35852611 (daylight) [104]
Subject 32236280 (challenging illumination) [101]
Subject 35697857 (complex surrounding) [102]
Multiple leopards [100]
Multiple wildlife [2]

Table A.2: Test Videos

A.4 A CALCULATION EXAMPLE OF THE MATCHING ALGORITHM



(a) SAM Masks

(b) Cutie Masks

(c) Putting all masks in one image

(d) Refined Masks

Figure A.1: A Calculation Example of the Matching Algorithm

$$\text{Mask IoU}: \begin{cases} \text{SAM1}: \begin{cases} \text{Cutie Mask 1}: & 0.4 \\ \text{Cutie Mask 3}: & 0.35 \\ \text{Cutie Mask 2}: & 0.1 \end{cases} \\ \text{SAM2}: \begin{cases} \text{Cutie Mask 2}: & 0.4 \end{cases} \end{cases}$$

$$\text{Inclusion SAM}: \begin{cases} \text{SAM1}: \begin{cases} \text{Cutie Mask 1}: & 0.4 \\ \text{Cutie Mask 3}: & 0.3 \\ \text{Cutie Mask 2}: & 0.1 \end{cases} \\ \text{SAM2}: \begin{cases} \text{Cutie Mask 2}: & 0.4 \end{cases} \end{cases}$$

$$\text{Inclusion Cutie}: \begin{cases} \text{SAM1}: \begin{cases} \text{Cutie Mask 1}: & 1 \\ \text{Cutie Mask 3}: & 0.9 \\ \text{Cutie Mask 2}: & 0.3 \end{cases} \\ \text{SAM2}: \begin{cases} \text{Cutie Mask 2}: & 1 \end{cases} \end{cases}$$

- **Step 1**: Sorting
  The SAM Masks are sorted based on their highest Mask IoUs. SAM Mask 1 and SAM Mask 2 both have the same highest Mask IoU, namely 0.4. In this case, the SAM Mask with the smaller mask value will be

first processed. Otherwise, the SAM Mask with a higher Mask IoU will be first processed.

The Cutie masks mapped to each SAM mask are also sorted in descending order based on their Mask IoUs with the SAM Mask.

- **Step 2**: Matching of SAM Mask 1
  Cutie mask 1, Cutie mask 2, and Cutie mask 3 are mapped to SAM mask 1.
  The sum of $InclusionRate_j^{\text{SAM}}$ ($j = 1, 2, 3$) by SAM mask 1 is 0.8.
  This sum is higher than the threshold 0.7, which means the SAM mask 1 is almost covered by its mapped Cutie masks. The Cutie masks might also be mapped to different SAM masks.

  Cutie mask 1 is processed first and must be processed because it has the highest Mask IoU with SAM mask 1. The sum of $InclusionRate_i^{\text{Cutie}}$ ($i = 1$) for Cutie mask 1 is 1 since the Cutie mask 1 is only mapped to the SAM mask 1 and totally included in the SAM mask 1. Cutie mask 1 is kept as an object mask.

  Cutie mask 3 is the mapped Cutie mask with the second highest Mask IoU with SAM mask 1. Cutie mask 3 is to be checked if it has already been processed or will be processed. This is not the case here. The Cutie mask 3 is kept as an object mask.

  Cutie mask 2 is the mapped Cutie mask with the lowest Mask IoU with SAM mask 1. Cutie mask 2 should also be checked if it has already been processed or will be processed. In this example, Cutie mask 2 has its highest Mask IoU with SAM mask 2, so Cutie mask 2 is ignored here and will be processed later when SAM mask 2 is processed.

- **Step 3**: Matching of SAM Mask 2
  Cutie mask 2 is mapped to SAM mask 2.

  The sum of $InclusionRate_j^{\text{SAM}}$ ($j = 2$) by SAM mask 2 is 0.4.
  The sum is lower than the threshold 0.7, which means the SAM mask 2 is not covered by its mapped Cutie mask. The SAM mask 2 should be added as a new object. Also, Cutie mask 2 must be kept as an object mask.

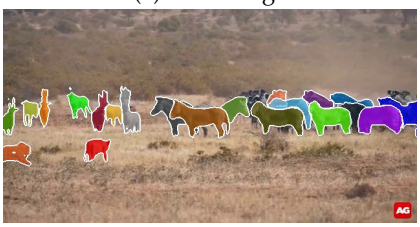
Note that the numbers in this example are not exactly calculated. They only describe approximately the relationship between the SAM masks and the Cutie masks.

A.5 QUALITATIVE PERFORMANCE OF THE MATCHING ALGORITHM

Matching results without any compromise, with the first compromising method, and with the second compromising method are demonstrated respectively.

(a) Raw image

(b) Cutie track masks



(c) SAM segmentation masks

(d) Refined masks without compromise

(e) Refined masks with compromising method 1

(f) Refined masks with compromising method 2

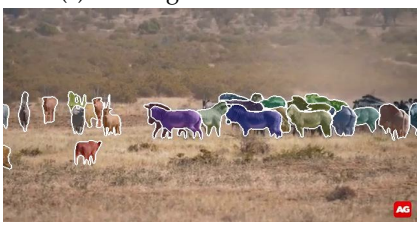
Figure A.2: 1*st* Update 30*th* Frame

(a) Raw image

(b) Cutie track masks

(c) SAM segmentation masks

(d) Refined masks without compromise



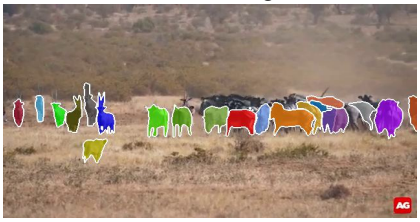
(e) Refined masks with compromising method 1

(f) Refined masks with compromising method 2

Figure A.3: 2*nd* Update 59*th* Frame

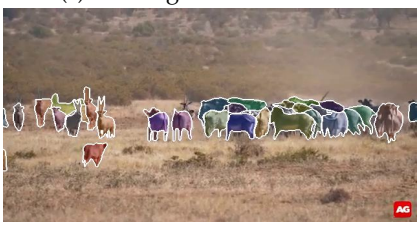
(a) Raw image

(b) Cutie track masks



(c) SAM segmentation masks

(d) Refined masks without compromise



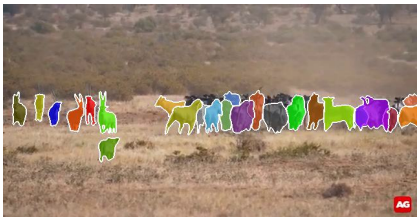
(e) Refined masks with compromising method 1

(f) Refined masks with compromising method 2

Figure A.4: 3*rd* Update 88*th* Frame

(a) Raw image

(b) Cutie track masks



(c) SAM segmentation masks

(d) Refined masks without compromise

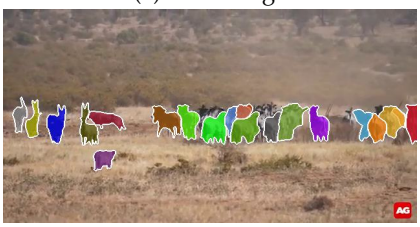
(e) Refined masks with compromising method 1

(f) Refined masks with compromising method 2

Figure A.5: 4*th* Update 117*th* Frame

(a) Raw image

(b) Cutie track masks



(c) SAM segmentation masks

(d) Refined masks without compromise

(e) Refined masks with compromising method 1

(f) Refined masks with compromising method 2

Figure A.6: 5*th* Update 146*th* Frame

BIBLIOGRAPHY

[1] Samira Abnar and Willem Zuidema. *Quantifying Attention Flow in Transformers*. 2020. arXiv: 2005.00928 [cs.LG].

[2] Africa Geographic. *Lioness hunts large zebra herd [Best Video Clip 2019 entry]*. [Online; accessed 12-Dezember-2023]. 2018. URL: https://www.youtube.com/watch?v=GKrP_ROLP2o.

[3] Phil Ammirato and Alexander C. Berg. *A Mask-RCNN Baseline for Probabilistic Object Detection*. 2019. arXiv: 1908.03621 [cs.CV].

[4] Microsoft Research Lab Asia. *Five reasons to embrace Transformer in computer vision*. https://www.microsoft.com/en-us/research/lab/microsoft-research-asia/articles/five-reasons-to-embrace-transformer-in-computer-vision/. [Online; accessed 19-Oktober-2023]. 2021.

[5] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. *End-to-End Attention-based Large Vocabulary Speech Recognition*. 2016. arXiv: 1508.04395 [cs.CL].

[6] Olivier Barnich and Marc Van Droogenbroeck. "ViBe: A Universal Background Subtraction Algorithm for Video Sequences." In: *IEEE Transactions on Image Processing* 20.6 (2011), pp. 1709–1724. DOI: 10.1109/TIP.2010.2101613.

[7] Rotimi-Williams Bello, E Ikeremo, Otobo Noah, Oc, Daniel Olubummo, and O Enuma. "Cattle Segmentation and Contour Detection Based on Solo for Precision Livestock Husbandry." In: *Journal of Applied Sciences and Environmental Management* 26 (Oct. 2022), pp. 1713–1720. DOI: 10.4314/jasem.v26i10.15.

[8] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. *YOLACT: Real-time Instance Segmentation*. 2019. arXiv: 1904.02689 [cs.CV].

[9] Rishi Bommasani et al. *On the Opportunities and Risks of Foundation Models*. 2022. arXiv: 2108.07258 [cs.LG].

[10] John Bongaarts. "IPBES, 2019. Summary for policymakers of the global assessment report on biodiversity and ecosystem services of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services." In: *Population and Development Review* 45.3 (2019), pp. 680–681. DOI: https://doi.org/10.1111/padr.12283. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/padr.12283. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/padr.12283.

[11] Thierry Bouwmans, Sajid Javed, Maryam Sultana, and Soon Ki Jung. "Deep Neural Network Concepts for Background Subtraction: A Systematic Review and Comparative Evaluation." In: *CoRR* abs/1811.05255 (2018). arXiv: 1811.05255. URL: http://arxiv.org/abs/1811.05255.

[12] COCO. https://cocodataset.org/#home. [Online; accessed 19-November-2023]. 2023.

[13] Caltech. https://lila.science/datasets/caltech-camera-traps. [Online; accessed 19-November-2023]. 2023.

[14] Rui Caseiro, Pedro Martins, João F. Henriques, and Jorge Batista. "A nonparametric Riemannian framework on tensor field with application to foreground segmentation." In: *Pattern Recognition* 45.11 (2012), pp. 3997–4017. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2012.04.011. URL: https://www.sciencedirect.com/science/article/pii/S0031320312001689.

[15] Gerardo Ceballos and Paul Ehrlich. "The Sixth Extinction Crisis Loss of Animal Populations and Species." In: *Journal of Cosmology* 8 (Nov. 2009).

[16] Tianrun Chen, Lanyun Zhu, Chaotao Ding, Runlong Cao, Yan Wang, Zejian Li, Lingyun Sun, Papa Mao, and Ying Zang. *SAM Fails to Segment Anything? – SAM-Adapter: Adapting SAM in Underperformed Scenes: Camouflage, Shadow, Medical Image Segmentation, and More*. 2023. arXiv: 2304.09148 [cs.CV].

[17] Bowen Cheng, Ross Girshick, Piotr Dollár, Alexander C. Berg, and Alexander Kirillov. *Boundary IoU: Improving Object-Centric Image Segmentation Evaluation*. 2021. arXiv: 2103.16562 [cs.CV].

[18] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. https://github.com/hkchengrex/Cutie. [Online; accessed 12-Dezember-2023]. 2023.

[19] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. *Putting the Object Back into Video Object Segmentation*. 2023. arXiv: 2310.12982 [cs.CV].

[20] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Alexander Schwing, and Joon-Young Lee. *Tracking Anything with Decoupled Video Segmentation*. 2023. arXiv: 2309.03903 [cs.CV].

[21] Ho Kei Cheng and Alexander G. Schwing. *XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model*. 2022. arXiv: 2207.07115 [cs.CV].

[22] Yangming Cheng, Liulei Li, Yuanyou Xu, Xiaodi Li, Zongxin Yang, Wenguan Wang, and Yi Yang. *Segment and Track Anything*. 2023. arXiv: 2305.06558 [cs.CV].

[23] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

[24] Simantika Choudhury, Nishant Bharti, Navajit Saikia, and Subhash Rajbongshi. "Detection of One-horned Rhino from Green Environment Background using Deep Learning." In: *Journal of Green Engineering* 10 (Oct. 2020), pp. 4657–4678.

[25] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. *What Does BERT Look At? An Analysis of BERT's Attention*. 2019. arXiv: 1906.04341 [cs.CL].

[26] Taco Cohen, Mario Geiger, and Maurice Weiler. *A General Theory of Equivariant CNNs on Homogeneous Spaces*. 2020. arXiv: 1811.02017 [cs.LG].

[27] iNaturalist Competition Datasets. https://github.com/visipedia/inat_comp/tree/master/2017. [Online; accessed 19-November-2023]. 2017.

[28] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. "Detecting moving objects, ghosts, and shadows in video streams." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25.10 (2003), pp. 1337–1342. DOI: 10.1109/TPAMI.2003.1233909.

[29] Hengfei Cui, Chang Yuwen, Lei Jiang, Yong Xia, and Yanning Zhang. "Bidirectional cross-modality unsupervised domain adaptation using generative adversarial networks for cardiac image segmentation." In: *Computers in Biology and Medicine* 136 (2021), p. 104726. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2021.104726. URL: https://www.sciencedirect.com/science/article/pii/S0010482521005205.

[30] Yan Dai, Ziyu Hu, Shuqi Zhang, and Lianjun Liu. "A survey of detection-based video multi-object tracking." In: *Displays* 75 (2022), p. 102317. ISSN: 0141-9382. DOI: https://doi.org/10.1016/j.displa.2022.102317. URL: https://www.sciencedirect.com/science/article/pii/S0141938222001354.

[31] wsyxbcl agentmorris Dan Morris. https://agentmorris.github.io/camera-trap-ml-survey/. [Online; accessed 16-November-2023]. 2023.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[33] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[34] Ahmed Elgammal, David Harwood, and Larry Davis. "Non-parametric Model for Background Subtraction." In: *Computer Vision — ECCV 2000*. Ed. by David Vernon. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 751–767.

[35] Shuqi Fang, Bin Zhang, and Jingyu Hu. "Improved Mask R-CNN Multi-Target Detection and Segmentation for Autonomous Driving in Complex Scenes." In: *Sensors* 23.8 (2023). ISSN: 1424-8220. URL: https://www.mdpi.com/1424-8220/23/8/3853.

[36] Mitchell Fennell, Christopher Beirne, and A. Cole Burton. "Use of object detection in camera trap image identification: Assessing a method to rapidly and accurately classify human and animal detections for research and application in recreation ecology." In: *Global Ecology and Conservation* 35 (2022), e02104. ISSN: 2351-9894. DOI: https://doi.org/10.1016/j.gecco.2022.e02104. URL: https://www.sciencedirect.com/science/article/pii/S2351989422001068.

[37] Swarnendu Ghosh, Nibaran Das, Ishita Das, and Ujjwal Maulik. *Understanding Deep Learning Techniques for Image Segmentation*. 2019. arXiv: 1907.06119 [cs.CV].

[38] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. *Masked Autoencoders Are Scalable Vision Learners*. 2021. arXiv: 2111.06377 [cs.CV].

[39] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. "Mask R-CNN." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[40] Tracey Hollings, Mark Burgman, Mary van Andel, Marius Gilbert, Timothy Robinson, and Andrew Robinson. "How do you find the green sheep? A critical review of the use of remotely sensed imagery to detect and count animals." In: *Methods in Ecology and Evolution* 9.4 (2018), pp. 881–892. DOI: https://doi.org/10.1111/2041-210X.12973. eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.12973. URL: https://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.12973.

[41] Yuhao Huang et al. *Segment Anything Model for Medical Images?* 2023. arXiv: 2304.14660 [eess.IV].

[42] M.-H Hung, Jeng-Shyang Pan, and C.-H Hsieh. "A fast algorithm of temporal median filter for background subtraction." In: *Journal of Information Hiding and Multimedia Signal Processing* 5 (Jan. 2014), pp. 33–40.

[43] Mao-Hsiung Hung, Jeng-Shyang Pan, and Chaur-Heh Hsieh. "Speed Up Temporal Median Filter for Background Subtraction." In: *2010 First International Conference on Pervasive Computing, Signal Processing and Applications*. 2010, pp. 297–300. DOI: 10.1109/PCSPA.2010.79.

[44] HungyiLee. *self-Attention*. https://speech.ee.ntu.edu.tw/~hylee/ml/ml2021-course-data/self_v7.pdf. [Online; accessed 09-December-2023]. 2021.

[45] Ramesh Jain and H.-H. Nagel. "On the Analysis of Accumulative Difference Pictures from Image Sequences of Real World Scenes." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.2 (1979), pp. 206–214. DOI: 10.1109/TPAMI.1979.4766907.

[46] Huaizu Jiang and Erik Learned-Miller. *Face Detection with the Faster R-CNN*. 2016. arXiv: 1606.03473 [cs.CV].

[47] Yasmin M. Kassim, Michael E. Byrne, Cristy Burch, Kevin Mote, Jason Hardin, David R. Larsen, and Kannappan Palaniappan. "Small Object Bird Detection in Infrared Drone Videos Using Mask R-CNN Deep Learning." In: *Electronic Imaging* 32.8 (2020), pp. 85–1–85–1. DOI: 10.2352/ISSN.2470-1173.2020.8.IMAWM-085. URL: https://library.imaging.org/ei/articles/32/8/art00003.

[48] Lei Ke, Yu-Wing Tai, and Chi-Keung Tang. *Deep Occlusion-Aware Instance Segmentation with Overlapping BiLayers*. 2021. arXiv: 2103.12340 [cs.CV].

[49] Asifullah Khan, Zunaira Rauf, Anabia Sohail, Abdul Rehman, Hifsa Asif, Aqsa Asif, and Umair Farooq. *A survey of the Vision Transformers and its CNN-Transformer based Variants*. 2023. arXiv: 2305.09880 [cs.CV].

[50] Wonjae Kim, Bokyung Son, and Ildoo Kim. *ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision*. 2021. arXiv: 2102.03334 [stat.ML].

[51] Alexander Kirillov et al. *Segment Anything*. 2023. arXiv: 2304.02643 [cs.CV].

[52] Shunfeng Li, Chunxue Wu, and Naixue Xiong. "Hybrid Architecture Based on CNN and Transformer for Strip Steel Surface Defect Classification." In: *Electronics* 11.8 (2022). ISSN: 2079-9292. DOI: 10.3390/electronics11081200. URL: https://www.mdpi.com/2079-9292/11/8/1200.

[53] Shilong Liu et al. *Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection*. 2023. arXiv: 2303.05499 [cs.CV].

[54] Wei Liu, Yuanzheng Cai, Miaohui Zhang, Hui Li, and Hejin Gu. "Scene background estimation based on temporal median filter with Gaussian filtering." In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. 2016, pp. 132–136. DOI: 10.1109/ICPR.2016.7899621.

[55] B.P.L. Lo and S.A. Velastin. "Automatic congestion detection system for underground platforms." In: *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No.01EX489)*. 2001, pp. 158–161. DOI: 10.1109/ISIMP.2001.925356.

[56] Alan M. "Background Subtraction Techniques." In: (Aug. 2001).

[57] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. *LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection*. 2016. arXiv: 1607.00148 [cs.AI].

[58] Nigel Mcfarlane and Charles Schofield. "Segmentation and tracking of piglets in images." In: *Machine Vision and Applications* 8 (Jan. 1995), pp. 187–193. DOI: 10.1007/BF01215814.

[59] MegaDetector. https://github.com/microsoft/CameraTraps/releases/tag/v5.0. [Online; accessed 12-Dezember-2023]. 2022.

[60] Jianbiao Mei, Mengmeng Wang, Yeneng Lin, Yi Yuan, and Yong Liu. *TransVOS: Video Object Segmentation with Transformers*. 2021. arXiv: 2106.00588 [cs.CV].

[61] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. *TrackFormer: Multi-Object Tracking with Transformers*. 2022. arXiv: 2101.02702 [cs.CV].

[62] MetaAI. https://dl.fbaipublicfiles.com/segment_anything/sam_vit_h_4b8939.pth. [Online; accessed 12-Dezember-2023]. 2023.

[63] Microsoft. https://www.microsoft.com/en-us/research/project/accelerating-biodiversity-surveys/. [Online; accessed 16-November-2023]. 2023.

[64] Microsoft. https://github.com/microsoft/CameraTraps/blob/main/megadetector.md#can-you-share-the-training-data. [Online; accessed 19-November-2023]. 2023.

[65] Harvard NLP. *The Annotated Transformer*. https://nlp.seas.harvard.edu/2018/04/03/attention.html. [Online; accessed 15-Oktober-2023]. 2018.

[66] Tapas Nayak and Hwee Tou Ng. *Effective Modeling of Encoder-Decoder Architecture for Joint Entity and Relation Extraction*. 2019. arXiv: 1911.09886 [cs.CL].

[67] James Parker. "A system for fast erosion and dilation of Bi-level images." In: *Journal of Scientific Computing* 5 (Jan. 1990), pp. 187–198. DOI: 10.1007/BF01089163.

[68] Lam Phan, Hiep Thi Hong Nguyen, Harikrishna Warrier, and Yogesh Gupta. "Patch Embedding as Local Features: Unifying Deep Local and Global Features Via Vision Transformer for Image Retrieval." In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. 2022, pp. 2527–2544.

[69] M. Piccardi. "Background subtraction techniques: a review." In: *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*. Vol. 4. 2004, 3099–3104 vol.4. DOI: 10.1109/ICSMC.2004.1400815.

[70] Stuart L. Pimm, Sky Alibhai, Richard Bergl, Alex Dehgan, Chandra Giri, Zoë Jewell, Lucas Joppa, Roland Kays, and Scott Loarie. "Emerging Technologies to Conserve Biodiversity." In: *Trends in Ecology Evolution* 30.11 (2015), pp. 685–696. ISSN: 0169-5347. DOI: https://doi.org/10.1016/j.tree.2015.08.008. URL: https://www.sciencedirect.com/science/article/pii/S0169534715002128.

[71] Pan African Programme. http://panafrican.eva.mpg.de/. [Online; accessed 16-November-2023]. 2023.

[72] Bo-Kai Ruan, Hong-Han Shuai, and Wen-Huang Cheng. *Vision Transformers: State of the Art and Research Challenges*. 2022. arXiv: 2207.03041 [cs.CV].

[73] Frank Schindler and Volker Steinhage. "Instance segmentation and tracking of animals in wildlife videos: SWIFT - segmentation with filtering of tracklets." In: *Ecological Informatics* 71 (2022), p. 101794. ISSN: 1574-9541. DOI: https://doi.org/10.1016/j.ecoinf.2022.101794. URL: https://www.sciencedirect.com/science/article/pii/S1574954122002448.

[74] Sanja Seljan, Tomislav Vičić, and Marija Brkic Bakaric. "BLEU Evaluation of Machine-Translated English-Croatian Legislation." In: May 2012. DOI: 10.13140/RG.2.1.4374.3204.

[75] Ajmal Shahbaz, Joko Hariyono, and Kang-Hyun Jo. "Evaluation of background subtraction algorithms for video surveillance." In: *2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*. 2015, pp. 1–4. DOI: 10.1109/FCV.2015.7103699.

[76] Yaser Sheikh, Omar Javed, and Takeo Kanade. "Background Subtraction for Freely Moving Cameras." In: *2009 IEEE 12th International Conference on Computer Vision*. 2009, pp. 1219–1225. DOI: 10.1109/ICCV.2009.5459334.

[77] Pierre-Luc St-Charles, Guillaume-Alexandre Bilodeau, and Robert Bergevin. "SuBSENSE : A Universal Change Detection Method with Local Adaptive Sensitivity." In: *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society* 24 (Dec. 2014). DOI: 10.1109/TIP.2014.2378053.

[78] Vanessa Suessle et al. *Automatic Individual Identification of Patterned Solitary Species Based on Unlabeled Video Data*. 2023. arXiv: 2304.09657 [cs.CV].

[79] Du-Ming Tsai and Shia-Chih Lai. "Independent Component Analysis-Based Background Subtraction for Indoor Surveillance." In: *IEEE Transactions on Image Processing* 18.1 (2009), pp. 158–167. DOI: 10.1109/TIP.2008.2007558.

[80] M. Van Droogenbroeck and O. Paquot. "Background subtraction: Experiments and improvements for ViBe." In: *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. 2012, pp. 32–37. DOI: 10.1109/CVPRW.2012.6238924.

[81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[82] Jesse Vig. *A Multiscale Visualization of Attention in the Transformer Model*. 2019. arXiv: 1906.05714 [cs.HC].

[83] Soroush Vosoughi, Prashanth Vijayaraghavan, and Deb Roy. "Tweet2Vec: Learning Tweet Embeddings Using Character-level CNN-LSTM Encoder-Decoder." In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. SIGIR '16. ACM, July 2016. DOI: 10.1145/2911451.2914762. URL: http://dx.doi.org/10.1145/2911451.2914762.

[84] Di Wang, Jing Zhang, Bo Du, Minqiang Xu, Lin Liu, Dacheng Tao, and Liangpei Zhang. *SAMRS: Scaling-up Remote Sensing Segmentation Dataset with Segment Anything Model*. 2023. arXiv: 2305.02034 [cs.CV].

[85] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. *SOLO: Segmenting Objects by Locations*. 2020. arXiv: 1912.04488 [cs.CV].

[86] Xiyu Wang, Pengxin Guo, and Yu Zhang. *Domain Adaptation via Bidirectional Cross-Attention Transformer*. 2022. arXiv: 2201.05887 [cs.CV].

[87] Qing Wu, Yungang Liu, Qiang Li, Shaoli Jin, and Fengzhong Li. "The application of deep learning in computer vision." In: *2017 Chinese Automation Congress (CAC)*. 2017, pp. 6522–6527. DOI: 10.1109/CAC.2017.8243952.

[88] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. *Zero-Shot Learning – A Comprehensive Evaluation of the Good, the Bad and the Ugly*. 2020. arXiv: 1707.00600 [cs.CV].

[89] Jinyu Yang, Mingqi Gao, Zhe Li, Shang Gao, Fangjing Wang, and Feng Zheng. *Track Anything: Segment Anything Meets Videos*. 2023. arXiv: 2304.11968 [cs.CV].

[90] Zongxin Yang and Yi Yang. *Decoupling Features in Hierarchical Propagation for Video Object Segmentation*. 2022. arXiv: 2210.09782 [cs.CV].

[91] Liangliang Yao, Haobo Zuo, Guangze Zheng, Changhong Fu, and Jia Pan. *SAM-DA: UAV Tracks Anything at Night with SAM-Powered Domain Adaptation*. 2023. arXiv: 2307.01024 [cs.CV].

[92] Jiangwei Yu, Xiang Li, Xinran Zhao, Hongming Zhang, and Yu-Xiong Wang. "Video State-Changing Object Segmentation." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, pp. 20439–20448.

[93] Ye Yu, Jialin Yuan, Gaurav Mittal, Li Fuxin, and Mei Chen. *BATMAN: Bilateral Attention Transformer in Motion-Appearance Neighboring Space for Video Object Segmentation*. 2022. arXiv: 2208.01159 [cs.CV].

[94] Lin Yuan and Zhao Qiu. "Mask-RCNN with spatial attention for pedestrian segmentation in cyber–physical systems." In: *Computer Communications* 180 (2021), pp. 109–114. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2021.09.002. URL: https://www.sciencedirect.com/science/article/pii/S0140366421003315.

[95] Muhamad Munawar Yusro, Rozniza Ali, and Muhammad Suzuri Hitam. "Comparison of Faster R-CNN and YOLOv5 for Overlapping Objects Recognition." In: *Baghdad Science Journal* 20.3 (2023), p. 0893. DOI: 10.21123/bsj.2022.7243. URL: https://bsj.uobaghdad.edu.iq/index.php/BSJ/article/view/7243.

[96] Mingyu Zhang, Fei Gao, Wuping Yang, and Haoran Zhang. "Real-Time Target Detection System for Animals Based on Self-Attention Improvement and Feature Extraction Optimization." In: *Applied Sciences* 13.6 (2023). ISSN: 2076-3417. DOI: 10.3390/app13063987. URL: https://www.mdpi.com/2076-3417/13/6/3987.

[97] Peng Zhang, Liucheng Hu, Bang Zhang, Pan Pan, and Alibaba. "Spatial Consistent Memory Network for Semi-supervised Video Object Segmentation." In: 2020. URL: https://api.semanticscholar.org/CorpusID:233404145.

[98] Z. Zivkovic. "Improved adaptive Gaussian mixture model for background subtraction." In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Vol. 2. 2004, 28–31 Vol.2. DOI: 10.1109/ICPR.2004.1333992.

[99] Zoran Zivkovic and Ferdinand van der Heijden. "Efficient adaptive density estimation per image pixel for the task of background subtraction." In: *Pattern Recognition Letters* 27.7 (2006), pp. 773–780. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2005.11.005. URL: https://www.sciencedirect.com/science/article/pii/S0167865505003521.

[100] Zoo Vienna Tiergarten Schönbrunn. *Leoparden-Nachwuchs im Tiergarten Schönbrunn*. [Online; accessed 12-Dezember-2023]. 2018. URL: https://www.youtube.com/watch?v=YyllU93bI_0.

[101] zooniverse. *Subject32236280*. [Online; accessed 12-Dezember-2023]. 2012. URL: https://www.zooniverse.org/projects/sassydumbledore/chimp-and-see/talk/subjects/32236280.

[102] zooniverse. *Subject35697857*. [Online; accessed 17-Dezember-2023]. 2017. URL: https://www.zooniverse.org/projects/sassydumbledore/chimp-and-see/talk/subjects/35697857.

[103] zooniverse. *Subject35698457*. [Online; accessed 12-Dezember-2023]. 2017. URL: https://www.zooniverse.org/projects/sassydumbledore/chimp-and-see/talk/2181/1369021.

[104] zooniverse. *Subject35852611*. [Online; accessed 12-Dezember-2023]. 2017. URL: https://www.zooniverse.org/projects/sassydumbledore/chimp-and-see/talk/2181/1647508.

[105] zooniverse. *Subject35718591*. [Online; accessed 12-Dezember-2023]. 2018. URL: https://www.zooniverse.org/projects/sassydumbledore/chimp-and-see/talk/2181/1466033.

[106] zooniverse. *Subject35857244*. [Online; accessed 12-Dezember-2023]. 2018. URL: https://www.zooniverse.org/projects/sassydumbledore/chimp-and-see/talk/2181/1569389?comment=2541535&page=1.