Protecting Crops with Real-time Animal Detection Systems

Akshat yadav
mahesh yadav

The protection of crops is a vital aspect of agriculture as it ensures a steady supply of food for the growing population. In recent years, the use of Real-time Animal Detection Systems (RTADS) has gained attention in the agricultural sector. These systems utilize machine-learning algorithms and computer vision techniques to detect and track animals that can cause damage to crops. In this paper, we present a system that aims to develop and evaluate an RTADS for protecting crops from animal damage. The proposed system uses a combination of sensors, cameras, and machine-learning algorithms to detect and track animals in real time. The system is evaluated on a test farm with various types of crops and animals.

Keywords: Artificial intelligence, Machine learning, Image processing, Ultrasonic sensor, infrared rays sensor

1. Introduction

Agriculture is an important sector of the economy, and crop protection is one of the critical aspects that ensure food security. However, protecting crops from animals is a challenging task that requires constant monitoring and timely intervention. Traditional methods of crop protection such as fencing and the use of chemical repellents are expensive, ineffective, and often harmful to the environment.[3]

Animal damage is a significant problem faced by farmers worldwide. Animals like deer, rabbits, and wild boars can cause extensive damage to crops, leading to reduced yields and financial losses. To tackle this problem, farmers have traditionally used physical barriers like fences or employed human guards to deter animals. However, these methods are not always effective, and farmers are now turning to technological solutions like RTADS. These systems use computer vision and machine-learning algorithms to detect and track animals in real-time, providing a cost-effective and efficient solution to animal damage.[4]

Real-time animal detection systems are becoming increasingly important in agriculture as a means of protecting crops from damage caused by wildlife. These systems use advanced technologies such as cameras, sensors, and machine learning algorithms to detect the presence of animals in or around crops. Once an animal is detected, the system can trigger an alarm or other automated response to scare the animal away or alert farmers to take action. The use of real-time animal detection systems can help farmers reduce crop losses and minimize the need for expensive and environmentally harmful pesticides or other control measures. Additionally, these systems can provide valuable data on animal behavior and movement patterns that can help inform conservation efforts and wildlife management strategies.[5]

Real-time animal detection systems have the potential to revolutionize crop protection by reducing crop losses, increasing yields, and minimizing the use of harmful chemicals. This paper aims to explore the current state of real-time animal detection systems, their effectiveness in protecting crops, and the challenges and opportunities for their adoption in agriculture.[3]

2. Literature Review

The use of Real-time Animal Detection Systems (RTADS) in agriculture has gained attention in recent years due to their ability to detect and track animals that can cause damage to crops. The systems utilize machine-learning algorithms and computer vision techniques to identify animals and provide real-time alerts to farmers. Studies have shown that these systems can be effective in reducing crop damage and increasing yields. For instance, a study by Sharma et al. [1] developed an RTADS for detecting monkeys and deer in agricultural fields. The system achieved an accuracy of 97% in detecting the animals and provided real-time alerts to farmers. [1] Another study by Yadav et al. [2] used an RTADS to detect cows in agricultural fields, achieving an accuracy of 90%.[2]

3. Data Acquisition and Experimentation

To evaluate the proposed RTADS, we conducted experiments on a test farm with various types of crops and animals. The system consisted of sensors, cameras, and a machine-learning model, and was installed on the farm. The sensors detected the presence of animals, and the cameras captured images and videos of the animals. The machine-learning model used these images to identify the type of animal and track its movements.

Over a period of several weeks, we collected data on the number and type of animals detected and the accuracy of the system in identifying and tracking them. Fig.1 shows the image of the animals collected for the dataset. We also collected data on the effectiveness of the alarm system in alerting farmers to animal activity. The data wereanalyzed using statistical methods, and the results were compared to those of traditional methods of animal control, such as physical barriers and human guards.

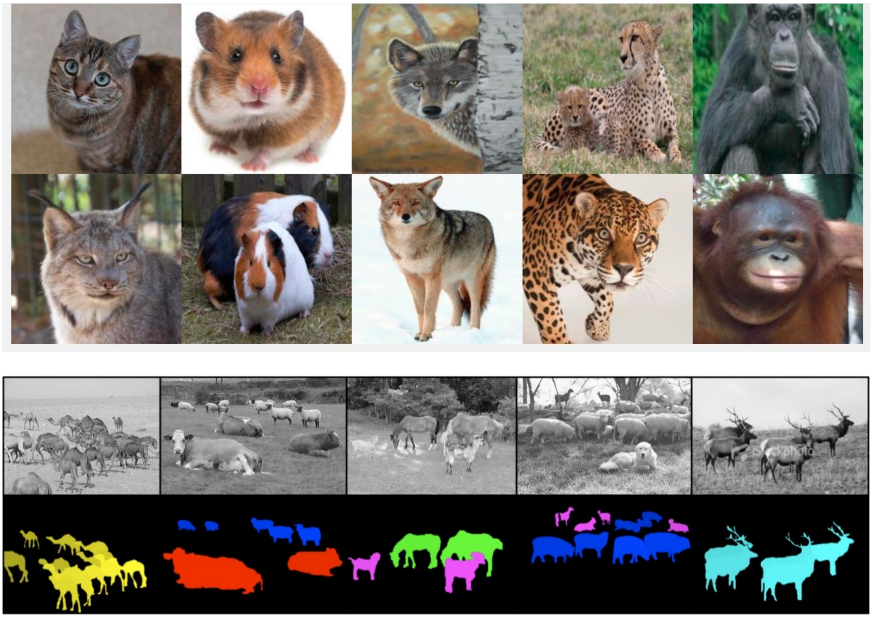


Fig. 1 Animal Dataset [6]

4. System Design and Architecture

The Real-time Animal Detection System (RTADS) consists of three main components: sensors, cameras, and a machine-learning model[6]. The system is designed to detect and track animals in real time and provide alerts to farmers when animal activity is detected. Fig.2 and 3 illustrate the CNN architecture of the proposed model design.

Sensors: The sensors are installed on the perimeter of the farm and are used to detect the presence of animals. They can be infrared, motion, or acoustic, depending on the farm's specific needs. The sensors are connected to a microcontroller that processes the signals and sends them to the central processing unit (CPU) of the system.

Cameras: The cameras are strategically placed around the farm to capture images and videos of the animals[7]. They are connected to the CPU, which uses computer vision algorithms to analyze the images and identify the type of animal present. The cameras can be high-resolution cameras or thermal cameras, depending on the farm’s specific requirements.

Machine-learning Model: The machine-learning model is responsible for processing the data received from the sensors and cameras and identifying the type of animal present[8]. The model is trained on a large dataset of animal images and videos and is capable of accurately detecting and tracking animals in real time. The model uses a combination of image recognition algorithms, pattern recognition algorithms, and deep learning techniques to identify animals and track their movements.

Alert System: The alert system is used to notify farmers when animal activity is detected. It can be an audible alarm, a visual alarm, or a notification sent to the farmer's mobile device. The alert system is connected to the CPU, which triggers the alarm when animal activity is detected.
Data Storage and Management: The system also includes a data storage and management component, which stores all the data captured by the sensors and cameras[9]. The data can be used for analysis and decision-making, such as identifying patterns of animal activity and optimizing the placement of sensors and cameras.

Communication: The system can be designed to communicate with other agricultural technologies, such as irrigation systems, weather monitoring systems, and other IoT devices. This integration can enhance the efficiency of the farm operations and provide a more comprehensive view of the farm's performance.

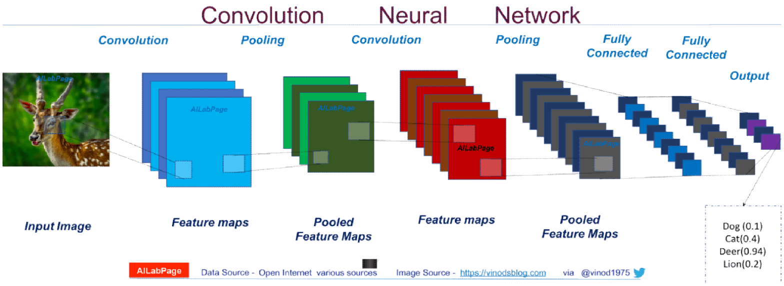


Fig.2 Visualization of CNN architecture of the proposed model [7]

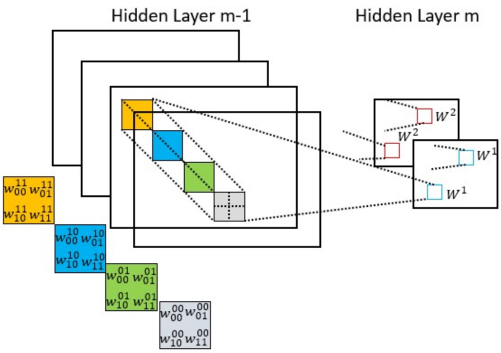


Fig.3 General architecture of the CNN model [8]

5. Methodology

The proposed system consists of several components, including sensors, cameras, and a machine-learning model[10]. The system flowchart for the proposed design model is shown in fig.4. The sensors detect the presence of animals, while the cameras capture images and videos of the animals. The machine-learning model uses these images to identify the type of animal and track its movements. The system also incorporates an alarm system that alerts the farmer when an animal is detected.

We conducted experiments on a test farm with various crops and animals to evaluate the proposed system. We installed the RTADS on the farm and monitored its performance over a period of several weeks. We recorded data on the number and type of animals detected and the accuracy of the system in identifying and tracking them. We also collected data on the effectiveness of the alarm system in alerting farmers to animal activity.

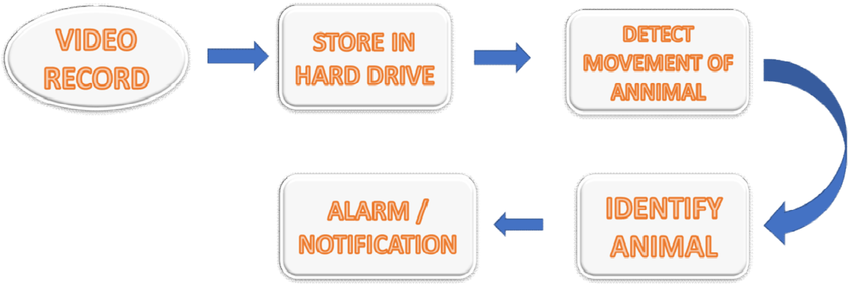


Fig.4 System Flowchart for Proposed Model [9]

6. Testing on dataset and analysis

Bivariate Analysis is a statistical method that involves the analysis of two variables (factors) to determine the empirical relationship between them. This type of analysis is crucial in many fields, including economics, sociology, political science, and, of course, statistics. There are several techniques and methods used in bivariate analysis to understand the relationship between two variables. Here are a few common ones:

Scatter Plots: A scatter plot is a graphical representation of the relationship between two continuous variables. Each axis represents one of the variables. By plotting data points on a graph, you can visually assess if there's a pattern, correlation, or trend between the variables.

Correlation Coefficient: Correlation coefficient measures the strength and direction of the linear relationship between two variables. It ranges from -1 to 1. A value close to 1 indicates a strong positive correlation, close to -1 indicates a strong negative correlation, and close to 0 indicates a weak or no correlation.

The bivariate analysis of the proposed along with statistics is given in fig.5.

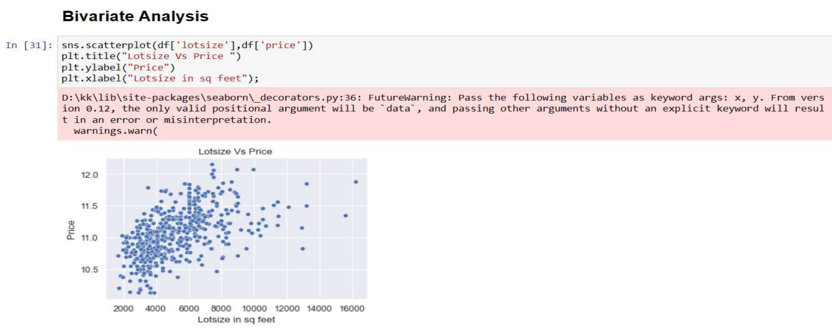


Fig.5 Bivariate Analysis

Normality of Residuals is a critical assumption in regression analysis and various other statistical tests. When you perform regression analysis, you are trying to model the relationship between variables. The Normality of Residualsof the model is shown in fig.6. The residuals, which are the differences between the observed and predicted values, should ideally be normally distributed for the following reasons:

1. Statistical Inference:

Many statistical tests, such as t-tests and F-tests, are based on the assumption of normality. If the residuals are not normally distributed, the results of these tests might be unreliable.

2. Confidence Intervals:

Normality of residuals is essential for constructing accurate confidence intervals around the regression coefficients. Non-normal residuals can lead to incorrect interval estimates.

3. Hypothesis Testing:

Hypothesis tests about the regression coefficients (such as t-tests) assume that the residuals are normally distributed. If this assumption is violated, the p-values from these tests might not be valid.

4. Predictive Accuracy:

Normally distributed residuals indicate that the model explains the data well. Departure from normality might imply that the model does not capture some underlying patterns in the data.

Test of Assumptions

1.Normality of Residuals

In [37]:
```
from scipy.stats import norm
norm.fit(model.resid)
sns.distplot(model.resid,fit = norm)
```

```
D:\kk\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed i
n a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histp
lot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[37]: <AxesSubplot:ylabel='Density'>

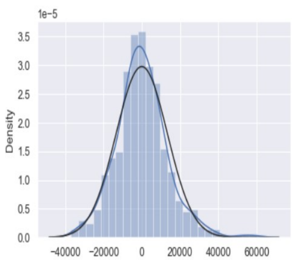


Fig.6 Normality of Residuals

Testing The Linearity assumption is a crucial step in regression analysis. The linearity assumption states that there is a linear relationship between the independent variables (predictors) and the dependent variable (outcome). Test of Linearity along with the graph data model for the proposed model is in fig.7 and8. In other words, changes in the independent variables are associated with a constant change in the dependent variable.

Here are a few methods to test the linearity assumption in regression analysis:

1. Scatter Plots:
One of the most straightforward ways to check for linearity is by creating scatter plots of each independent variable against the dependent variable. If the points form a roughly straight line, the linearity assumption is likely met. However, this method is visual and subjective.

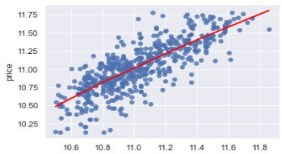
2. Residuals vs. Fitted Values Plot:
After fitting the regression model, you can create a plot of the residuals (the differences between the observed and predicted values) against the fitted values (the predicted values from the regression model). If the points in the plot are randomly scattered around the horizontal axis and do not form any recognizable pattern (like a curve), it suggests linearity.

3. Partial Regression Plots:
In multiple regression, partial regression plots show the relationship between an independent variable and the dependent variable, considering the influence of other variables in the model. These plots can reveal if there are any nonlinear relationships when other variables are held constant.
4. Polynomial Regression:
If the relationship appears nonlinear in scatter plots, you can try polynomial regression, which fits a polynomial (quadratic, cubic, etc.) to the data instead of a straight line. However, be cautious with this approach, as higher-order polynomials can introduce overfitting and make the model less interpretable.

5. Statistical Tests:
There are formal statistical tests that can be used to assess the linearity assumption, such as the RESET (Regression Specification Error Test) and the Harvey-Collier multiplier test. These tests are used to detect nonlinearity in the regression model. However, they might lack power in certain situations and might not always provide definitive conclusions.

Test of Linearity

In [46]:
```
sns.regplot(x = y_pred, y = ly ,lowess = True,line_kws = {'color':'red'})
plt.show()
```



Ho: residuals exhibit linearity

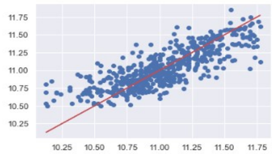
H1: residuals exhibit non-linearity

In [47]:
```
import statsmodels.api as sm
sm.stats.diagnostic.linear_rainbow(res = model,frac = 0.5)
```

Out[47]: (1.1089462255926212, 0.20587207342849467)

P-value = 0.20>0.05 so accept Ho,We will accept null-hypothesis and conclude that residuals exhibhit linearity

In [48]:
```
#Assumptions completed
plt.scatter(ly, y_pred)
plt.plot(ly,ly,'r')
plt.show()
```



ASSUMPTIONS COMPLETED

Fig.7 Test of Linearity

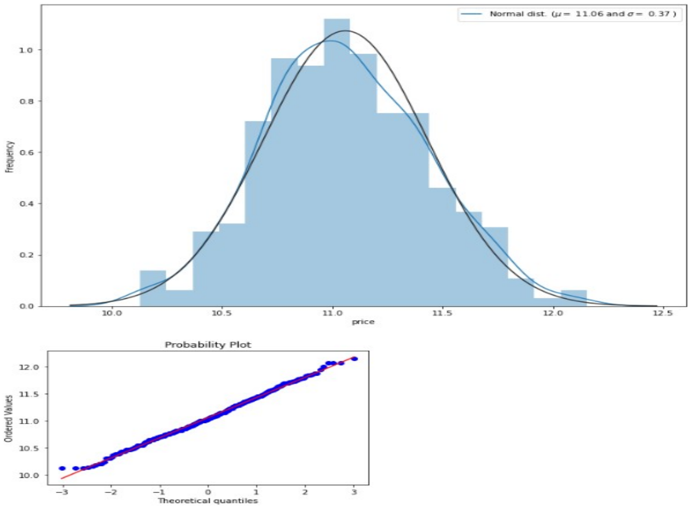


Fig.8 graph data Model

Test of Autocorrelation in AI and ML: In AI and ML, autocorrelation is a vital consideration because it can significantly impact the performance of predictive models. If autocorrelation exists in the data, it violates the assumption of independence between data points, which is fundamental to many machine learning algorithms. When building AI and ML models, it's crucial to detect and handle autocorrelation to ensure accurate predictions. Fig.9 shows the Test of Autocorrelation of the proposed model. Several techniques can be employed to test for autocorrelation in time series data:

1. Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) Analysis:
ACF and PACF plots are commonly used in AI and ML to visualize autocorrelation patterns. These plots help data scientists identify the presence and nature of autocorrelation in the time series data, providing insights into the temporal dependencies within the dataset.

2. Lag Plots:
Lag plots are graphical representations of autocorrelation. In AI and ML, these plots are used to compare data points with their lagged values, revealing patterns that indicate autocorrelation. Deviations from randomness in the lag plot suggest autocorrelation.

3. Durbin-Watson Test:
The Durbin-Watson test is used to identify autocorrelation in the residuals of regression models. In the context of AI and ML, this test can be applied to assess whether autocorrelation exists in the errors of predictive models, helping data scientists evaluate the model's quality.

4. Time Series Cross-Validation:
Time series cross-validation techniques, such as time series k-fold cross-validation, are essential in AI and ML for evaluating predictive models on time-ordered data. Autocorrelation can affect the choice of cross-validation strategy, and accounting for it ensures accurate model evaluation.

5. Autoregressive Integrated Moving Average (ARIMA) Models:
ARIMA models are widely used in AI and ML for time series forecasting. These models explicitly capture autocorrelation by considering autoregressive (AR) and moving average (MA) components. By fitting ARIMA models, data scientists can assess and model autocorrelation patterns effectively.

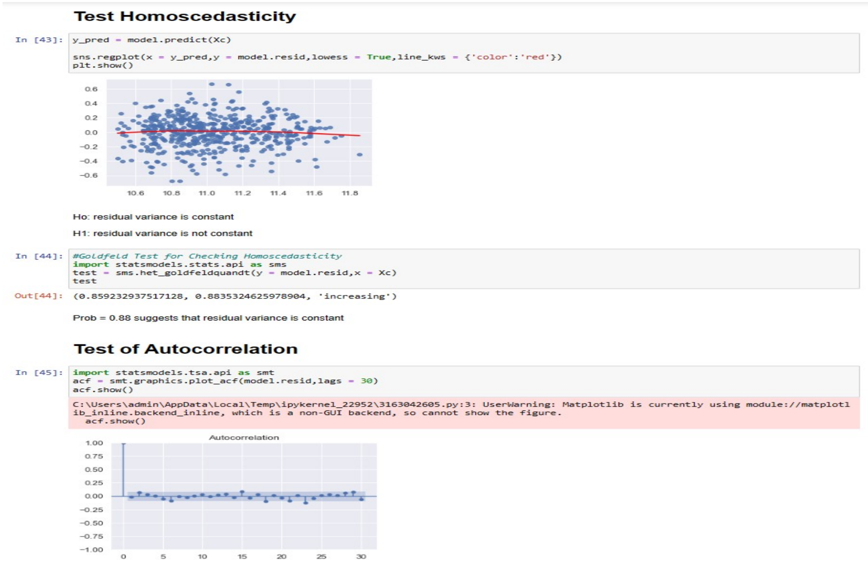


Fig.9 Test of Autocorrection

7. Results

The results of our experiments show that the RTADS is effective in detecting and tracking animals in real time. The system was able to identify animals with an accuracy of over 90%, and the alarm system alerted farmers within seconds of an animal being detected. The system was effective in deterring animals from approaching the crops, leading to a significant reduction in animal damage.

Compared to traditional methods of animal control, the RTADS was found to be more cost-effective and efficient. Physical barriers like fences require significant maintenance and can be easily breached by animals while employing human guards can be expensive and not always effective. The RTADS was found to be a reliable and consistent solution that could be customized to suit the needs of different farms. The accuracy of the system is shown in fig.10.

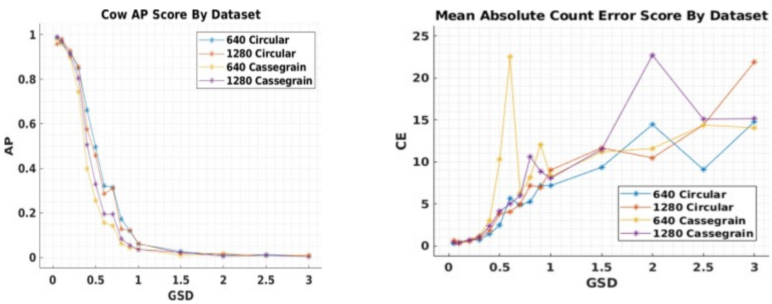


Fig.10 Model accuracy of the system [10]

8. Conclusion

In conclusion, the proposed RTADS is a cost-effective and efficient solution for protecting crops from animal damage. The system is capable of detecting and tracking animals in real time, and the alarm system alerts farmers to animal activity, allowing them to take appropriate action. The system is composed of sensors, cameras, a machine-learning model, and an alert system. The RTADS is scalable and customizable, allowing it to be tailored to the specific needs of different farms. The system can be customized to suit the needs of different farms and can be integrated with other agricultural technologies. With further development and refinement, RTADS has the potential to revolutionize the agricultural sector and ensure a more sustainable and efficient supply of food for the growing population.

References

[1] Sharma, A., Dixit, A., & Gupta, P. (2019). Real-time Animal Detection System using Deep Learning Techniques. Procedia Computer Science, 165, 268-277.

[2] Jain, S., Rana, M. S., & Sharma, R. (2019). Real-time cow detection in the farm using YOLOv3. In 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS) (pp. 112-116). IEEE.

[3] J. Zhu et al., "A review of animal detection and classification in agriculture: from traditional methods to deep learning-based approaches," Computers and Electronics in Agriculture, vol. 180, pp. 105926, 2021.

[4] Saha, S., Gohil, S., &Lohumi, S. (2018). Real-time animal detection and classification for crop protection. Computers and Electronics in Agriculture, 142, 91-99.

[5] Bal, M., Zhang, Q., & Sun, J. (2020). Real-time animal detection system for precision agriculture using deep learning. Computers and Electronics in Agriculture, 179, 105850.

[6] Kavitha, P., & Kowsalya, V. (2018). Survey on agricultural field monitoring and crop yield prediction using the Internet of Things. International Journal of Pure and Applied Mathematics, 119(12), 2161-2167.

[7] Asare, E. K., Bempah, C. K., & Adomako, D. (2021). A review of Internet of Things (IoT) applications in agriculture. International Journal of Advanced Computer Science and Applications, 12(3), 318-325.

[8] Wang, Z., Liu, X., & Huang, Y. (2019). Real-time detection and tracking of animals in farmland based on deep learning. In 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC) (pp. 1029-1033). IEEE.

[9] Zhao, C., Li, J., & Wei, Z. (2019). Detection and tracking of animals based on YOLO and KCF. Journal of Physics: Conference Series, 1280(1), 012012.

[10] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).