BirdsEye: Enabling Rapid Supervised Image Dataset Creation for Geospatial-AI

Morgan Masters, *Member IEEE*, Adam Korycki, *Member IEEE*, T. Luca Altaffer, *Member IEEE*, Nick Kuipers, Nick Bender, Colleen Josephson, *Member IEEE*, Steve McGuire, *Member IEEE*

***Abstract*—Modern robotic perception systems depend on large, high-quality labeled datasets; in outdoor environments, collecting and annotating such data remains prohibitively labor-intensive. This bottleneck limits the scalability of AI systems in domains like agriculture, ecology, and infrastructure monitoring. We present *BirdsEye*, a geospatial annotation pipeline that enables rapid, semi-automated dataset creation using uncrewed aerial vehicles, Real-Time Kinematic Global Positioning System data, onboard cameras, and field-deployable handheld devices. By tightly coupling precise projective calibration with human-guided annotation, *BirdsEye* reduces the time and expertise required to generate labeled training data for visual learning models.**

In a case study of pest burrow detection in agricultural fields, *BirdsEye* produced over 55,000 geotagged annotations across more than 12,500 images in the equivalent of a single workday of field labor. A convolutional neural network trained on this data achieved 83.7% recall on a geographically distinct test site, demonstrating generalization to unseen terrain. Compared to traditional GUI-based labeling, *BirdsEye* yields a 300%-500% increase in net annotation throughput, while maintaining high spatial accuracy and requiring drastically fewer workers. Through efficient, accurate data collection in real-world settings, *BirdsEye* offers a democratized path forward for deploying learning-based perception systems in agriculture and other outdoor domains.

***Index Terms*—Geospatial annotation, geospatial AI, dataset generation, field robotics, RTK-GPS, computer vision, projective calibration, agricultural automation, human-in-the-loop systems.**

I. Introduction

Field robotics offers powerful platforms for monitoring vegetation, wildlife, and terrain in remote or inaccessible environments [1], [2], [3], [4], [5], [6], [7], [8]. While deep convolutional neural networks (CNNs) can automate the interpretation of imagery they fetch, their performance hinges on the availability of reliable datasets to learn patterns that humans recognize intuitively.

Accessing the power of machine learning (ML) and object recognition in field environments requires a large corpus of annotated data labeling features of interest. However, generating large, labeled datasets remains a significant challenge, forcing research communities to rely on large-scale, internet-hosted datasets. These datasets are often curated by major AI developers, such as Google (e.g., Open Images [9]) and Amazon (e.g., AWS Open Data [10]), or through large-scale crowd-sourcing initiatives [11], [12], [13], [14], [15], [16].

While these online repositories have been instrumental in advancing AI, their content is shaped by the priorities of sponsoring organizations and the expertise of contributors to crowd-sourced efforts. Consequently, researchers and practitioners working in specialized domains—such as field robotics for environmental monitoring or agricultural automation—often find that their specific data needs are underrepresented [17].

For instance, numerous well-annotated plant health datasets exist, with sizes ranging from several hundred to tens of thousands of labeled examples [18], [19], [20], but these are predominantly captured from close-up perspectives. A farmer attempting to autonomously manage a 500-acre (2 km^2) operation, however, requires data views from practical perspectives for effective field assessment.

There is an abundance of aerial and satellite imagery available to the public and a corresponding battery of geospatial annotation tools (see Section II-C2). These sources of imagery have spatial resolutions as fine as 30 cm, in the case of high-end paid services by providers like Airbus and Maxar Technologies. However, most free-to-access satellite imagery sources, like the Sentinel and Landsat constellations, have spatial resolutions in range of 10-60 m. Similarly, modern aircraft can capture imagery at spatial resolutions under 10 cm, based on product datasheets from sensor producers like MicaSense/AgEagle, Teledyne FLIR, Leica, and Vexcel. While foundational in many fields, these spatial resolutions are inadequate to capture the finest details of ecological and agricultural systems, where target features exist at the scale of millimeters to a few centimeters. Moreover, while ML-based super-resolution techniques can artificially enhance apparent image detail, they are inappropriate for learning real-world features in scientific contexts, as they risk introducing hallucinated structures not present in the original scene.

Terrestrial imaging, whether performed by humans or ground robots, can achieve extremely high spatial resolution, but manual post-processing the returned imagery is time-prohibitive at the scale of data produced and navigation is constrained by traversability. In contrast, low-altitude UAV imaging offers a far more time-efficient alternative, capturing high-resolution data across broad areas in a fraction of the time. For example, a UAV flying at 10 meters AGL with a 1920×1200 camera and a 30° horizontal field of view (HFoV) lens can resolve features down to 2 mm while skipping the difficulties of terrestrial navigation.

The principal barrier to applications of UAVs to ultra-fine

M. Masters, A. Korycki, N. Kuipers, C. Josephson, and S. McGuire are affiliated with the Department of Electrical and Computer Engineering, UC Santa Cruz, CA 95060 USA

N. Bender and T. L. Altaffer are graduated Masters of Science students of the Department of Electrical and Computer Engineering, UC Santa Cruz, CA 95060 USA

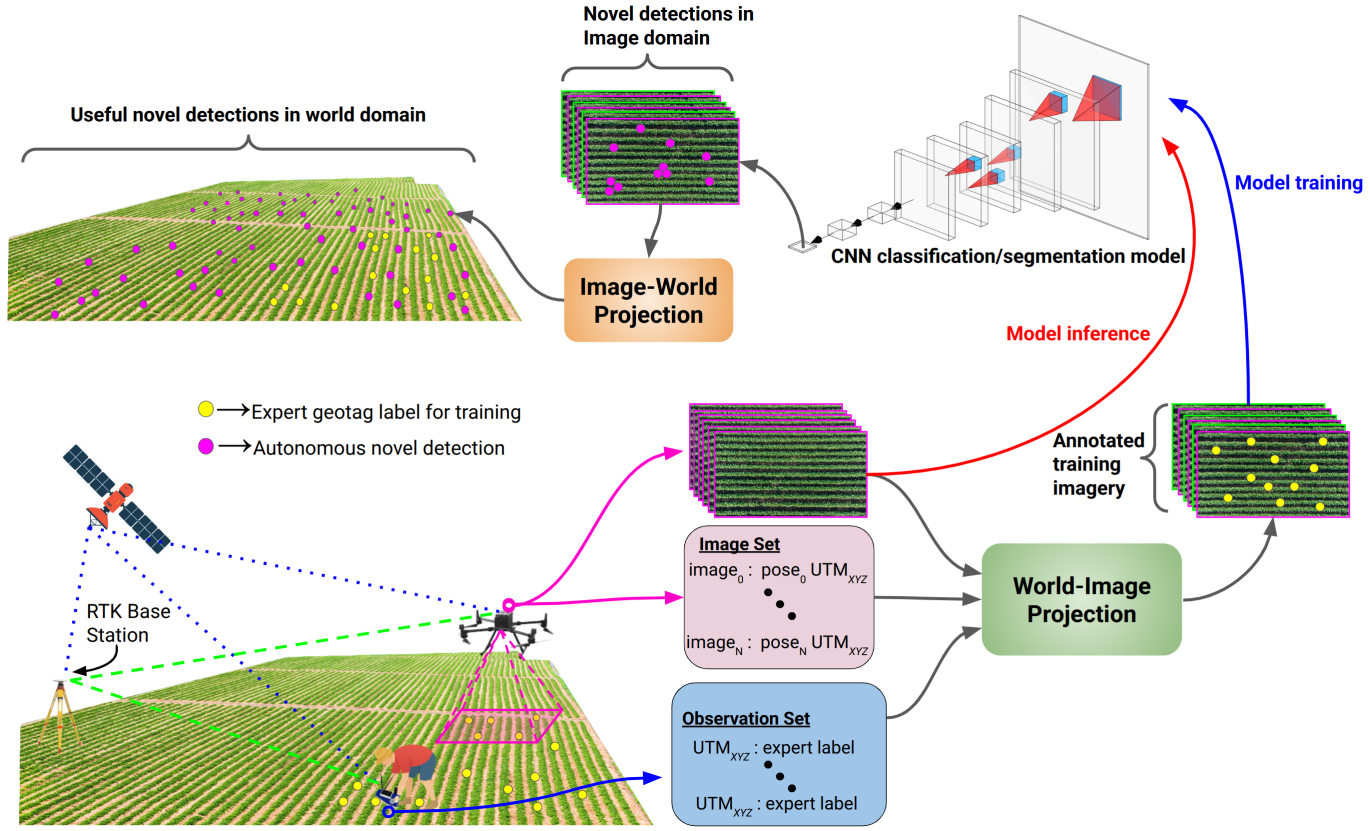


Fig. 1. The *BirdsEye* UAV-based image annotation system. Pictured (bottom left) are the specialist in the field with a handheld annotation device, the RTK GPS base station, and the UAV surveying a field. System software (bottom right) then either collates expert annotations and UAV imagery via camera projective models, creating an annotated dataset, or passes UAV imagery directly to a trained neural model for feature detection and subsequent georeferencing of detection event pixels (top left).

feature tracking is an absence of relevant labeled datasets from their unique low-altitude aerial viewpoint. Generating high-quality, annotated training data for such applications remains time-consuming, labor-intensive, and expensive [21]. Thus, dataset creation represents a critical bottleneck, preventing practitioners from leveraging deep learning methods to enhance decision-making and operational efficiency.

In response, we introduce *BirdsEye*, a UAV-based data ingestion, annotation, and training platform designed to assist non-experts in AI with training, maintaining, and deploying custom deep learning models for outdoor applications. *BirdsEye* significantly reduces the effort required for aerial image annotation by replacing frame-by-frame manual image labeling with structured, on-the-ground observations from subject matter experts. As depicted in Figure 1, the first stage of the *BirdsEye* system involves using a handheld device to create geotags in the field. For example, a field worker could geotag instances of diseased plants in an otherwise healthy farm field.

During the next phase, a UAV autonomously flies over the geotagged field and collects environmental imagery. The images are then sent through a processing pipeline which correlates the geotags to pixel regions in the imagery. This results in a fully supervised training dataset, which can then be used to train a vision model.

Finally, the UAV can be used as an automatic surveyor, where all detection events recorded throughout a flight video are georeferenced using the inverse of the projective map we used to annotate the imagery. Returning to our agricultural example, the labeled images might be used to train a model to classify the plants as healthy versus diseased. Once the model is trained, it can be used in the field, without human input, to map all spotted occurances of the training target in a world reference frame (e.g. Google Earth).

Our key innovation is a semi-automated approach to training field-deployable computer vision models using geotagged feature observations. These observations consist of the Real-Time Kinematic Global Positioning System (RTK GPS) coordinates of geotags, indicating an instance of a training target, the target's class, and other relevant metadata. When combined with UAVs equipped with RTK GPS, these annotations can be automatically detected within image frames and used to generate training data efficiently. This approach significantly reduces the burden of data annotation while enabling rapid deployment of domain-specific models in resource-limited or dynamic environments. To demonstrate the system's impact in reducing the annotation bottleneck, we present a case study of automatically detecting pest burrows in commercial farmland.

However, the deployment of autonomous systems in agricultural settings entails significant ethical responsibility. Tools like *BirdsEye* must not be viewed solely through the lens of efficiency. While our system demonstrates that large-scale environmental annotation is feasible, such capabilities must

be governed with transparency, oversight, and respect for labor rights and ecological impact. Data-driven models cannot substitute for the judgment of domain experts, and so human-in-the-loop validation must remain a core tenet of these workflows.

In summary, our key contributions include:

- A novel approach to minimizing effort in outdoor supervised training dataset construction,
- Characterization of the error behavior of the *BirdsEye* system, and
- A case study demonstrating the use and deployment of the *BirdsEye* system.

The source code and example CAD files (made to interface with a DJI M300-RTK) are publicly available at https://github.com/harelab-ucsc/birdseye.

II. Background and Related Work

A variety of annotation tools have been developed to alleviate the burden of manual image labeling, leveraging scalable human-in-the-loop approaches. Among these, two dominant paradigms have emerged: large-scale crowd-sourced annotation (e.g., reCAPTCHA, Sec. II-A) and paid microtask platforms (e.g., Amazon Mechanical Turk, Sec. II-B).

Beyond these, numerous commercial services offer professional annotation solutions tailored to domain-specific applications (Sec. II-C). However, these systems primarily focus on post-collection annotation, lacking integration with real-time, geospatially-aware dataset creation. This limitation significantly impacts applications requiring high-precision outdoor training data grounded in expert knowledge, such as agriculture, autonomous navigation, and environmental monitoring.

A. Crowd-Sourced Annotation: reCAPTCHA and Passive Labeling

Google's reCAPTCHA system has evolved into a societally pervasive passive annotation framework, embedding image labeling tasks into routine web interactions [22]. By leveraging human responses for CAPTCHA validation, reCAPTCHA generates large-scale labeled datasets at minimal cost. However, the system's reliance on arbitrary task assignment by the service provider precludes its use for domain-specific dataset construction. The average user is likely unable to accomplish tasks like identifying disease symptoms on specific plants.

B. Paid Microtask Annotation: Amazon Mechanical Turk (MTurk)

Amazon Mechanical Turk (MTurk) provides an on-demand, crowd-based annotation marketplace, enabling dataset generation through distributed microtasks [23]. The platform has been widely used for image classification, object detection, and natural language processing due to its scalability and cost-efficiency. However, since task execution is highly dependent on worker expertise, rigorous validation protocols become necessary.

C. Limitations of Existing Annotation Frameworks

Automated annotation systems have played a crucial role in advancing supervised learning by enabling large-scale dataset construction. The primary categories of annotation tools we consider are: general-purpose annotation platforms, geospatial imagery labeling frameworks, and domain-specific, synthetic dataset generators for robotics and autonomous navigation. Each category presents fundamental constraints that hinder their applicability to high-precision outdoor dataset collection.

1) General-Purpose Annotation Tools: Several cloud-based platforms, including Labelbox, CVAT, and Roboflow, provide scalable solutions for image annotation and dataset management [24], [25], [26]. These tools integrate AI-assisted labeling techniques, collaborative workflows, and API-based automation to accelerate dataset generation. Despite their versatility, these systems are often fundamentally manual annotation strategies with AI support for problems already solved by larger AI models.

2) Geospatial Imagery Annotation Tools: Specialized platforms, such as Picterra and LabelMe, cater to aerial and satellite imagery annotation [27], [28]. These systems leverage georeferenced imagery to enable spatially aware annotations, making them well-suited for remote sensing applications. However, they are predominantly designed as post-processed image analysis, adding annotations to pre-collected datasets, rather than using field annotations to automatically label imagery.

3) Synthetic Datasets: For those seeking an alternative to hardware-dependent data collection methods like *BirdsEye*, synthetic data generation offers a scalable solution. Rendering engines such as Unreal Engine [29] enable the rapid creation of photorealistic datasets [30], [31], [32], [33], [34], which are particularly valuable for early-stage prototyping in robotics and computer vision. However, bridging the domain gap between synthetic and real-world data remains a significant challenge, as models trained on synthetic imagery often struggle to generalize to real-world conditions. Fundamentally, this is due to failures to uphold objective realism (in terms of textures, lighting behaviors, materials properties, and other physical properties) in simulation, resulting in unrealistically clean training data.

D. Open-Source Datasets

A wide array of publicly available outdoor datasets have been instrumental in advancing AI research [11], [12], [13], [14], [15], [16], [18], [35]. These datasets have been foundational to state-of-the-art AI advancements [36], [37], [38], [39]. Major datasets even exist in the aerial imagery domain [40], [41], [42], providing pre-collected imagery from the UAV perspective.

In practice, their applicability is often constrained by their predominant focus on structured, urban environments and highly-funded problems (e.g. self-driving vehicles and pedestrian tracking). This bias limits their usefulness for practitioners working in diverse, unstructured outdoor settings. BirdsEye aims to bridge this gap by enabling users to generate high-quality datasets of arbitrary subject matter on demand.

E. Pixel Georeferencing

Georeferencing is the task of assigning geospatial coordinates to pixels in aerial or satellite imagery. Accurate georeferencing is critical for applications in environmental monitoring and autonomous navigation, and particularly for ultra-fine feature georeferencing tasks (features on the sub-cm scale). Two dominant paradigms exist: Ground Control Point (GCP)-based methods and direct georeferencing.

1) GCP-Based Georeferencing: The traditional approach to georeferencing uses GCPs, or visually identifiable landmarks with high-accuracy ground-truth coordinates. GCPs are distributed in the survey area and, by establishing correspondences between image pixels and their known ground coordinates, a transformation model (e.g. projective or homography-based) is estimated to map the image plane to a geospatial reference system. [43] reports an accuracy of 3.5 cm when using 10-12 GCPs per km^2. However, the ground sampling distance (GSD) in their study was 1.7 cm making this infeasible for ultra-fine grain feature tracking. The core issue with this approach is that deploying dense GCPs is impractical in agricultural settings, where data infrastructure is forced to be unobtrusive by heavy agricultural machinery.

2) Direct Georeferencing: Direct georeferencing offers a contactless alternative to traditional ground control point (GCP)-based methods by eliminating the need for surveyed visual markers. Instead, it relies on onboard localization sensors to estimate the camera's six degree-of-freedom pose at the precise moment of image capture. This typically involves tightly-coupled GNSS/IMU integration to provide accurate position and orientation estimates. When paired with a calibrated camera model, this enables pixel-level projection to and from a global reference frame (e.g., North-East-Down).

Liu et al. [43] report an absolute georeferencing accuracy of 8.7 cm using direct methods, while [44] finds a limiting performance of 11 cm $\pm$ 2.14 cm. However, both studies employ imagery with relatively coarse GSDs, on the order of several centimeters, which limits their ability to resolve small-scale surface features. In contrast, direct georeferencing applied to imagery with sub-3 mm GSD—as in high-resolution low-altitude UAV platforms—requires significantly higher precision to avoid pixel-level misalignment. At this scale, even centimeter-level georeferencing errors can result in multiple-pixel displacements, making sensor calibration, synchronization, and intrinsic accuracy critical to successful implementation. Moreover, both studies relied on proprietary georeferencing software, limiting reproducibility and posing barriers to low-cost, open-source implementations. In contrast, *BirdsEye* implements a fully-local, open-source direct georeferencing platform aimed at low-altitude, ultra-high resolution feature tracking (<3mm GSD).

III. Design and Methods

The *BirdsEye* platform consists of both hardware (UAV, annotation hardware) and software that work together to produce labeled image datasets. The following sections present the design and methods of our hardware and software stacks, our field operations protocol, as well as the design of our case study AI model.

A. The BirdsEye Hardware Platform

The core of our hardware system is a consumer-grade UAV equipped with a visual-inertial sensing payload and RTK GPS receivers (see Figure 3). Our UAV platform is a DJI Matrice M300 RTK and we are using a single Raspberry Pi 5 as our sensor payload's computer, with ROS2 as our system coordination software. In addition to the UAV and payload, we present our geoannotation system.

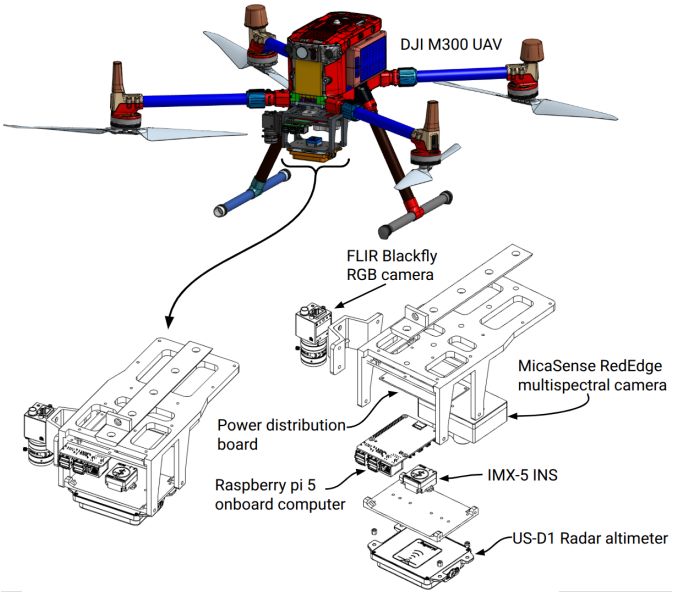


Fig. 2. The *BirdsEye* hardware platform based on the DJI M300-RTK UAV. The payload includes RGB and multispectral visual, inertial, GNSS+RTK, and radar altimetry sensing modalities. All components are mounted to a lightweight 3D printed frame.

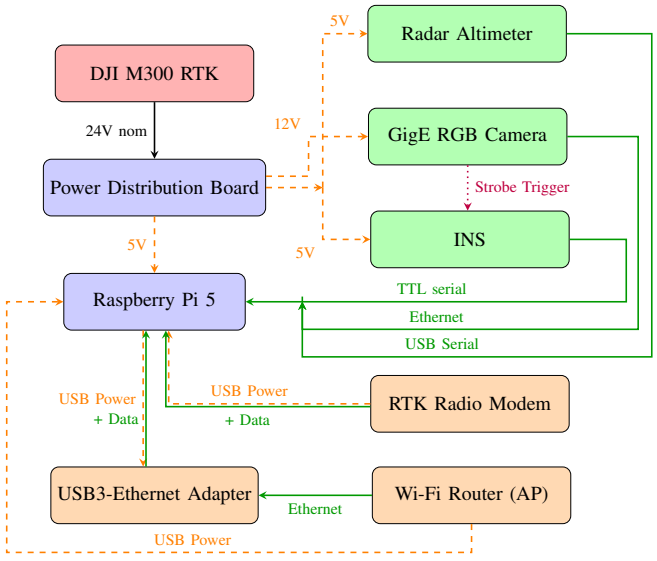


Fig. 3. System diagram of the onboard robotic payload integrated with the DJI M300 RTK.

1) Payload Sensors: The *BirdsEye* framework is built to support several visual sensing modalities. Foremost, we use the `Aravis` library[1] to drive Gigabit Ethernet (GigE) machine

[1]https://aravisproject.github.io/aravis/

vision cameras. In aerial robotics, where consistent data flow and minimal latency are critical, GigE cameras provide greater resilience than USB3 by decoupling image capture from host CPU activity, as well as improving mechanical robustness and electrical isolation in a high shock and vibration environment. By using a ROS2 middleware, we can support a wide variety of cameras, including those leveraging UVC-compliant or vendor-specific USB transports, as well as other Ethernet-connected cameras that use custom transports.

In our case, we utilize a combination of FLIR Blackfly RGB GigE cameras, a FLIR Boson 640 long-wave infrared (LWIR) camera with a UVC USB interface, and the MicaSense RedEdge-MX, a standard multispectral imager used in agricultural surveys with a custom Ethernet interface. In Figures 2 and 3, we illustrate the basic layout of the hardware system. Figure 3 includes only the RGB camera to reflect the system configuration used in our case study.

In addition to the visual sensing suite, we include a radar altimeter and an RTK GPS-provisioned inertial navigation system (INS). The INS fuses internal magnetometer, barometer, accelerometer, and gyroscope data with RTK positioning for high-fidelity inertial estimates of pose. Our device is an InertialSense RUG-3-IMX-5-RTK, connected via a hardware serial port in order to eliminate nondeterminism inherent to a USB serial interface.

We used COLMAP [45] estimates of Brown-Conrady camera model intrinsic and distortion parameters, computed from the raw imagery of a calibration dataset. Then, we provided those parameters and the inertial data captured to the Kalibr visual-inertial calibration process [46] to estimate the rigid body transform between our system's IMU and cameras.

2) Geoannotation System: Although conventional smartphone GNSS receivers support global localization, their positional accuracy, limited to the meter-level, is inadequate for resolving fine-grained terrestrial features. Consequently, the *BirdsEye* system employs real-time kinematic (RTK) positioning, augmented via a wireless correction network, to achieve centimeter-level accuracy. The geoannotation system we have constructed features an RTK basestation and hand-portable rover units. Our basestation consists of an RTK GPS receiver (u-blox Zed F9P), a radio modem (RFD900x with point to multipoint firmware), and an 801.11n 2.4ghz Wi-Fi access point (AP), designed to accommodate future mesh networking research. The basestation surveys its position on startup and, upon acquiring a specified error in estimated position, broadcasts RTCM3 corrections. Additionally, the Wi-Fi access point broadcasts a local NTRIP server, to provision a more general set of RTK-compatible wireless devices.

The rover units are handheld annotation devices, featuring the same GPS and radio equipment as the basestation and a Wio Terminal[2] for user interface (Figure 4). This device allows a user to specify geotag metadata for the features they plan to survey for, as well as providing real-time RTK connectivity details to the user. A user can geotag any environmental feature by aligning the device's antenna above it and triggering a position capture. Geotag positions and metadata are appended to a `*.csv` file on an SD card, for easy data portability.

Fig. 4. Handheld geo-annotation device being used by a partnering farmer interested in gopher-mitigation applications of the *BirdsEye* system.

B. Field Operations Protocol

Each deployment starts with powering the RTK base station and the annotation devices, then allowing the base station to survey in to an RTK fix and for RTK fix statuses on all handhelds. While this occurs, the environment is partitioned into small areas which are easily single-person surveyable in roughly an hour. Upon establishment of a fix on all RTK network devices, the field partitions are surveyed sequentially by a team of annotators. When a partition's survey completes, records of feature locations are offloaded from the devices to a shortest-path flight planner script, powered by the `fast-tsp` python library[3], and then the drone is launched on the resulting autonomous flight mission (capturing imagery at 5Hz).

Fig. 5. An illustration of the variance within the general class of "burrow". (Left) An obvious burrow with a small flattened mound. (Center) An occluded burrow and an obvious mound; is "mound" as important as "hole"? (Right) A very old, flattened mound with no visible hole; how degraded a target is allowable?

1) Defining Field Survey Inclusion Criteria: At the outset of each field exercise in dataset construction, it is crucial to have a consistent inclusion criterion specified, so that annotators generate labels of consistent noise levels. See Figure 5 for an example of the ambiguities of defining the training target. This underscores the value of specified heuristics to

[2]https://www.seeedstudio.com/Wio-Terminal-p-4509.html

[3]https://fast-tsp.readthedocs.io/en/latest/

guide surveys; controlling this ambiguity through agreed-upon heuristics and ground personnel training prevents us from sending vague training signals to learning models.

2) Restriction to Annotated Spaces: It is critical that the drone is kept from flying through regions of the deployment environment which annotators did not cover. This, and also instances where annotators failed to spot targets during their survey, generates false negatives in the resulting dataset.

In controlling these two factors from the start of the field operation, users mitigate the impact of noise in the labeling scheme and the resulting training labels. While the flight paths of the drone are easily controlled by structured surveys, radiating from the home point of the drone, the more subtle issue of features which the annotators missed can be mitigated in software postprocessing (Section III-D).

C. The BirdsEye Software Backend

After in-field collection, the data are sent through an offline batch postprocessing step, compatible with consumer laptops. The software backend runs geotag annotation or detection processes (Figure 6). It consists of a time synchronization step, a data collation step, and a database streaming step.

The time synchronization step takes advantage of our hardware triggering harness between the camera and the INS to extract the pose corresponding to the beginning of each image's exposure. During the data collation step, the data are parsed from raw sensor streams into data frames which are stored in a SQLite3 database. Finally, in the streaming step, we use a geometric detection method to associate geotags with their pixel coordinates in every frame they populate or run an inference model on the imagery and georeference detections. We describe each of these steps in detail below.

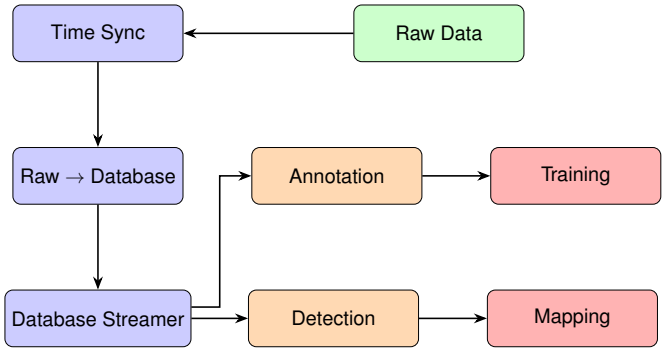


Fig. 6. The *BirdsEye* software pipeline including visual-inertial time synchronization via hardware strobing, *sqlite* database population, and geotag-to-pixel projection.

1) Visual-Inertial Time Synchronization: We use the `Aravis` library[4] and our GigE camera to source a hardware strobe pulse upon the start of each in-flight camera exposure, to enable sufficient alignment in time between pose captures and image captures. This strobe trigger passes to the onboard INS, which immediately captures a 6-DoF pose with a bitfield flag indicating that it was triggered by a strobe input. These poses are easily segregated, then synchronization with imagery is a simple alignment of the pose and image sets with two heuristics to eliminate false matches.

The first heuristic is that the difference in age between the pose and its matching image should never exceed the reciprocal of the framerate. If this occurs, we receive a new pose after the capture period and discard the preceding pose. The second heuristic is that the best match is the one with the minimum time difference. Upon finding a match, we remove the paired image from the set of remaining candidates, to avoid double-matches. This combination of hardware support and processing heuristics allows us to reduce latency induced by the image transport mechanism to be on the order of logic delays (ns).

2) Data Collation: Once parsed from a ROS2 data stream into our SQLlite database, sensor data can be analyzed offline without requiring live connections to the robotic system. Our parser subscribes to multiple ROS2 topics, collating image frames with correlated metadata such as altimeter readings, positional data, and training feature geotag positions. These are synchronized by timestamp and structured into discrete time-step frames within the database. This unified dataset enables efficient temporal and multimodal analysis, facilitating downstream tasks like mapping, object detection, and system diagnostics.

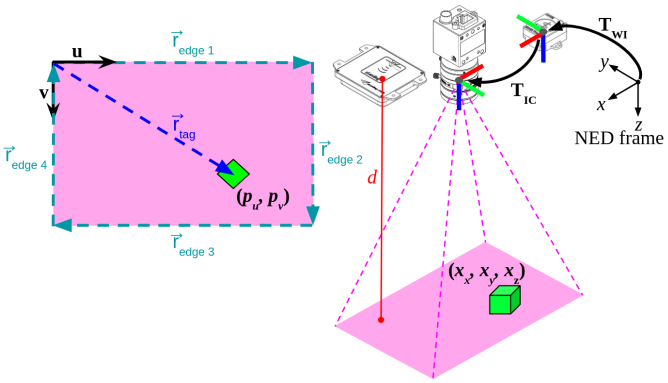


Fig. 7. Illustration of the geometric relationship between a 3D geotag in the world and its 2D projection in the image. Also pictured are the position vectors relevant to the cross-products in the geotag detection method of Section III-C3.

The core mechanism of the *BirdsEye* system is correlating the GPS ground annotations to the corresponding pixels in the UAV-collected imagery. We use a calibrated Brown-Conrady projective camera model, which enables world-to-image transformations of the annotations via rigid-body and projective transforms, illustrated in Figure 7. A geotag $\mathbf{x} = (x_x, x_y, x_z)$ in the NED (north-east-down) world frame is observed by a camera rigidly mounted to an INS. The transformations $\mathbf{T}_{\text{WI}}$ (world to INS) and $\mathbf{T}_{\text{IC}}$ (INS to camera) define the full pose chain from the world frame to the camera frame. The geotag is then projected to a 2D pixel coordinate $\mathbf{p} = (p_u, p_v)$ via $\mathbf{K}$, the camera's intrinsic matrix (estimated by COLMAP in Section III-A1):

$$\mathbf{p} = \mathbf{K}\mathbf{T}_{\text{IC}}\mathbf{T}_{\text{WI}}\mathbf{x}$$

[4]https://github.com/AravisProject/aravis

Scalar depth d is obtained from a radar altimeter measurement and is assumed constant across the entire image. We are working in farm fields, where such an assumption is a practical shortcut around digital terrain modeling, though ongoing work is developing 3D reconstruction replacements for this simple altimetry model, based upon photogrammetry and interpolated radar altimetry. This process allows backprojection of each image pixel to a corresponding point on the ground. For rigorous discussions of the process of constructing the transform chain, from fundamental sensor parameters to sensor-to-sensor rigid body transforms, see [47], [48]. All data (geotag positions and metadata, georeferenced image data frames, and relevant sensor calibration parameters) is saved to a set of tables in a portable SQLite3 database file.

3) Geotag-to-Pixel Annotation: We project a cone, whose hull is defined by the image frame's boundary in undistorted pixel space, to the approximated surface of the UAV's environment. In our case, we are assigning annotation depth d based on the readings of our radar altimeter. Any geotags within this cone indicate that a training target is in view. The inclusion check progresses clockwise about the frame's bounding rectangle, in pixel units. For each segment of the boundary, we compute the cross-product of the segment and each projected geotag's pixel coordinates. For example, let $\overrightarrow{r}_{edge} = [x_{edge}, y_{edge}]^T$ be the displacement vector between 2 adjacent vertices of the frame boundary and $\overrightarrow{r}_{tag} = [x_{tag}, y_{tag}]^T$ be the position vector of a projected geotag (these vectors are illustrated in Figure 7). Then:

$$\overrightarrow{r}_{edge} \times \overrightarrow{r}_{tag} = x_{edge}y_{tag} - x_{tag}y_{edge}.$$

The sign of the resulting value indicates which side of the line segment the geotag is on; positive if the geotag is in the right halfspace of the segment. If this quantity is positive for a clockwise walk through all bounding rectangle segments, then the corresponding geotag is in-frame. This process hinges on centimeter-scale localization, made possible through RTK GPS data at both the handheld device and the camera payload.

D. Using BirdsEye Annotated Imagery for AI Model Training

The output of the *BirdsEye* system is a high-fidelity labeled dataset of UAV imagery. These images can be used to train an arbitrary AI/ML model, which then can be used for in-field inference (e.g. detection of pest burrows or diseased plants). To demonstrate the efficacy of our annotation system, we created an example dataset and model, in a case study on using *BirdsEye* for gopher hole detection.

We parse our annotated frames to 224x224 subtiles, which places greater emphasis on small-scale, local features. This emphasis supports better learning for classes with subtle differences from the background class, like a gopher hole atop bare soil or radiometric differences between healthy and unhealthy plants in multispectral imagery. Labels are specified as isotropic Gaussian humps, centered on the projected pixel coordinates of in-frame geotags and with an adjustable parameter controlling the width of the label, σ (set to 10 pixels in our case study). These Gaussian humps are drawn out to a 3-σ radius. In practice, our soft label patches measure 60 pixels (corresponding to 30 cm, in projected units) in diameter.

Our example model is a neural network with a pretrained EfficientNetV2B3 backbone feature extractor, using weights pretrained on ImageNet, provided by Google through the Tensorflow Keras API. Atop the backbone, we implemented a deep upsampling head, designed to return the backbone's feature map to the source imagery's resolution via learned upsampling with strided, transposed convolution kernels. The model is first trained to an early stopping criterion with the backbone's weights frozen, then fine-tuned by unfreezing the top 30% of the model's layers and restarting training. See Table I for relevant details on the case study model's architecture. We use standard binary cross-entropy loss to train the model, and train the model using an Nvidia 3070 GPU, up to a termination criterion which waits for 15 epochs of no improvement in validation loss before stopping the process. This takes roughly 18 hours.

TABLE I
MODEL SUMMARY, WHERE DATA FLOWS SEQUENTIALLY FROM INPUTLAYER THROUGH CONV2D, IN ROW-ORDER.

Layer (type)	Output Shape	Param #
InputLayer	(None, 224, 224, 3)	0
EfficientNetV2-B3	(None, 7, 7, 1536)	12,930,622
Conv2DTranspose	(None, 14, 14, 512)	7,078,400
Conv2DTranspose	(None, 28, 28, 256)	1,179,904
Conv2DTranspose	(None, 56, 56, 128)	295,040
Conv2DTranspose	(None, 112, 112, 64)	73,792
Conv2DTranspose	(None, 224, 224, 32)	18,464
Conv2D	(None, 224, 224, 1)	33
Total params		21,576,255 (82.31 MB)
Trainable params		15,249,745 (58.17 MB)
Non-trainable params		6,326,510 (24.13 MB)

We apply class balancing because our dataset is dominated by negative examples upon converting frames to tiles by discarding purely negative tiles according to a randomized scheme. In this scheme, we retain 20% of negative tiles by randomly sampling from a uniform distribution, ensuring a rough class balance.

In addition to balancing the positive and negative classes of our dataset, this randomized subsampling of the negative class offers us an escape from instances where a target feature was not annotated by field personnel - a problem mentioned in Section III-B2. In the event of a false negative instance being left behind during field annotation, it still has high probability of being dropped from the training dataset (80%, in our implementation). This implicit error removal step is highly valuable; the elimination of false annotations is at least as impactful on the overall performance of the training process as the addition of data, in the data-centric AI view [49].

IV. RESULTS

In what follows, we first present our characterization of the projective performance of the system in Section IV-A. Then, we discuss our dataset generation case study and quantitative comparisons between *BirdsEye* and manual annotation in Section IV-B. Lastly, we present a performance characterization of our case study's trained detector in Section IV-C.

TABLE II
NORMS AND UNCERTAINTIES OF ERROR VECTORS FROM GEOREFERENCING BENCHMARK TESTS AT DIFFERENT FLIGHT MANEUVERS AND ALTITUDES, REPORTED IN METERS AND PIXELS.†

Altitude	Flight Maneuver	Error Norm		Standard Deviation	
		Meters	Pixels	Meters	Pixels
10m	Hovering	**0.038**	18.8	**0.019**	9.7
	Yawing	0.191	82.2	0.052	22.1
	Head-on	0.045	19.6	0.024	10.4
	Strafing	0.062	27.4	0.044	19.7
20m	Hovering	0.043	9.8	0.037	8.4
	Yawing	0.179	41.1	0.065	14.8
	Head-on	0.081	18.6	0.033	7.6
	Strafing	0.074	16.9	0.069	15.7

†The data of this table gives estimates of our effective GSD at both test altitudes: 0.002219±0.000138 m (2.219±0.138 mm) @ 10 m; 0.004376±0.000021 m (4.376±0.0212 mm) @ 20 m.

A. Characterization of Projection Error

The accuracy of the projection of the 3D geotag in the world onto 2D images is a critical component of the *BirdsEye* system. Projection errors are primarily introduced by miscalibration, effecting the quality of geospatial annotations and degrading downstream model performance. For example, even projection errors of as little as 60 pixels, corresponding to a 20 cm 3D offset, can place the training label of a geotag sufficiently off-target that we accidentally train on false positive features.

To quantify system projective accuracy, we designed a validation procedure using geotagged AprilTag targets in a flat, level environment. This setup provides both pixel-space ground truth via AprilTag detections and world-frame ground truth via RTK-corrected geotag positions, enabling comprehensive evaluation of both 2D and 3D projection fidelity.

1) Experimental Setup and Flight Maneuvers: Validation was conducted through a series of flight maneuvers at altitudes of 10 m and 20 m, designed to emulate common operational behaviors. These included static hovers and yawing rotations, as well as nose-on and strafing overflights. Images were captured at 5 Hz for at least 120 seconds per maneuver, yielding approximately 600 frames per maneuver iteration. Each maneuver was repeated five times, resulting in over 3,000 data points per maneuver category.

2) Projection Accuracy: Figure 8 illustrates the spatial distribution of projection errors in both world coordinates (top) and pixel coordinates (bottom), while Table II summarizes the system's projection performance using the calibration parameters deployed throughout the remainder of our experiments. The geospatial projection error remained within single-digit centimeters in both mean and standard deviation, excluding flights involving sustained yawing.

In all cases, we see that the standard deviation of projections of points into imagery is less than 23 pixels. Under the most favorable conditions (hovering at 10 m), we observed sub-2 cm standard deviation corresponding to a $\mu + 2\sigma$ confidence boundary (95.4% credible interval), of 38.2 pixels or 3.18% of the image frame's minor axis. In the most challenging case (yawing at 10 m), this interval increased to 126.4 pixels, or 10.53%. A corollary result of Table II is an estimate

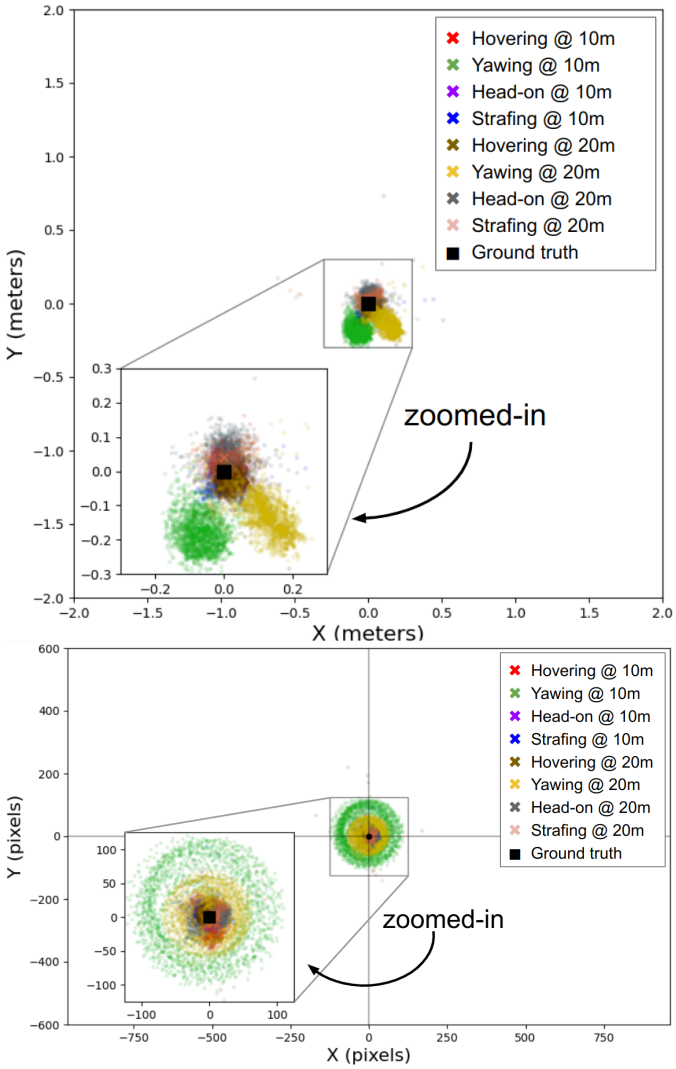


Fig. 8. (TOP) The distributions of system target projections in the real world, projected on the horizontal plane. (BOTTOM) The distributions of system target projections in imagery. Note that the axes of this panel are scaled to the actual shape and size of a captured frame. Insets in each figure provide expanded views of the projection clusters in both pixel and metric coordinate systems.

of our GSD at 10- and 20-meter test altitudes. *BirdsEye* attains 2.2 mm GSD at 10 m and 4.4 mm GSD at 20 m. These results suggest that the calibration procedure is sufficient for accurate pixel georeferencing in typical flight conditions. In our autonomous flight plan, we constrained the drone to point north in order to eliminate the effect of high error associated with sharp yawing maneuvers.

B. Dataset Generation and Annotation Efficiency

A primary motivation for creating *BirdsEye* is to reduce the burden of aerial image annotation. To evaluate the efficiency of the annotation pipeline, we conducted a field campaign at three sites within the region of Santa Cruz, CA (Figure 9). Datasets were constructed using RGB imagery captured via autonomous flights, following the protocols of Section III-B. Table III summarizes the flight metadata and dataset composition.

TABLE III
SUMMARY OF VALIDATION FLIGHT DATES, LOCATIONS, AND DATASET COMPOSITIONS. THE UPPER TABULAR BLOCK PRESENTS OUR FLIGHTS TO BUILD THE MAIN BODY OF OUR CASE STUDY DATASET, WHILE THE LOWER BLOCK PRESENTS THE RESULTS OF OUR TIMING EXPERIMENTS.

Flight ID	Date	Location	Dataset Composition		
			# Geotags	# Frames	Annotations
F1	2025-04-04	UCSC Farm	1,181	10,034	15,000
F2	2025-04-09	UCSC Farm	308	6,514	10,578
F3	2025-04-16	Pie Ranch	161	2,120	6,354
F4	2025-04-21	UCSC Farm	256	3,371	8,117
F5	2025-04-23	Jacobs Farm / del Cabo	181	2,555	6,613
F6	2025-04-23	UCSC Farm	151	3,647	11,715
F7	2025-05-02	Jacobs Farm / del Cabo	390	5,440	8,490
Totals			*2,628*	*33,681*	*66,867*

Date	Dataset Composition			Dataset Build Time		
	# Geotags	# Frames	Annotations	Field Work	Postprocessing	Total Time
2025-05-21	999	3,976	25,315	2:07:17 (2)†	0:14:27	4:29:01
2025-05-23	1,015	4,910	10,573	1:44:33 (2)†	0:19:29	3:48:35
2025-05-26	1,303	3,638	19,712	1:49:27 (2)†	0:12:06	3:51:00
Totals	*3,317*	*12,524*	*55,600*			*12:08:36*

†The multiplier in parentheses represents the number of workers active. The labor hours of the exercise is the product of the time result and the worker multiplier.

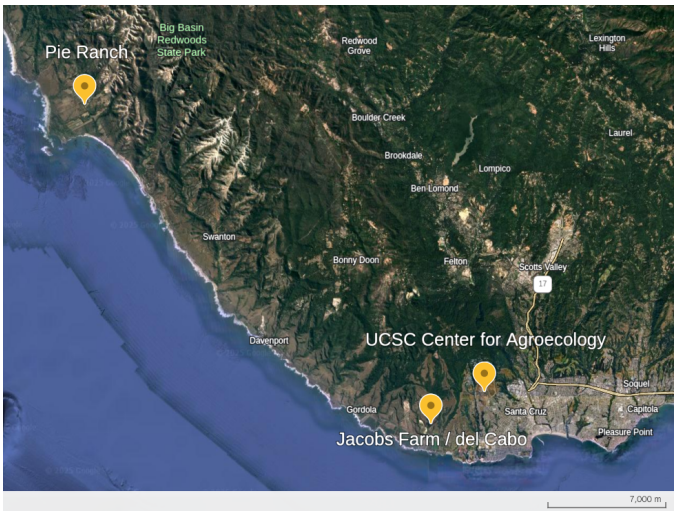


Fig. 9. Map of sites at which we collected training footage, including partnering commercial farms, Pie Ranch and Jacobs Farms / del Cabo, and the UCSC Center for Agroecology. Image source: Google Earth.

TABLE IV
COMPARISON OF THE RATES OF WORK FOR HUMAN ANNOTATORS USING TRADITIONAL ANNOTATION STRATEGIES, VERSUS *BirdsEye*. *BirdsEye* FRAME COUNT AND FIELD WORK TIME IS TRANSCRIBED FROM THE LOW BLOCK OF TABLE III.

	Workers	Frames	Labor Hours	Rate (s/frame)↓
Manual	15	1,500	4:48:52	11.55
BirdsEye	2	12,524	11:22:52	**3.27**

Annotation rate comparisons against 15 non-specialist annotators, each tasked with annotating 100 random frames from our dataset using GUI-based methods, are shown in Table IV. Compared to traditional annotation, *BirdsEye* achieves a 300%-500% increase in annotation throughput. Moreover, it took only 2 people to achieve this rate increase, compared to the 15 people in the baseline group. This corresponds to an increase of single-worker productivity of 2200%-3800%.

C. Model Training and Performance Evaluation

To demonstrate a simple use case for rapid aerial annotation, we created a prototype model to assist in detecting pest burrows. Burrowing pests are estimated to cause over $168 million in damages annually on California agriculture alone [50]. Our burrow detection model is a CNN trained using 44,085 annotated images from two sites, with evaluation performed on a geographically distinct test set (Flight F3, Pie Ranch). The model outputs heatmaps in the image plane, which are binarized to generate discrete detection events. Each detection is projected into the world frame and clustered using DBSCAN [51] to mitigate noise.

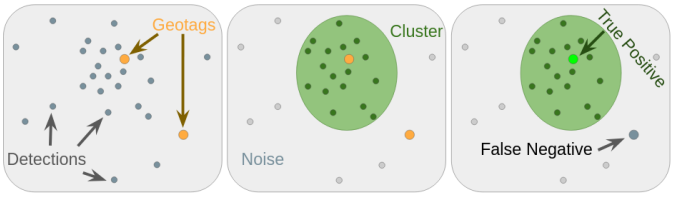


Fig. 10. (Left) A detection pointcloud (grey dots) near 2 geotags. (Center) DBSCAN clustering and denoising of the detection pointcloud. (Right) Nearest-neighbor association test between geotags and detection clusters. If a geotag is close enough to the core of a DBSCAN cluster, then it is counted as a true positive detection, else it is a false negative. Likewise, if a cluster exists and does not include a geotag, then it is a false positive detection.

1) Evaluation Metric: In order to develop a sense of model performance, we define comparison metrics between sparse human geotag annotations and denser model detection event pointclouds. By "pointclouds", we refer to the sets of 3D points laid out by the human annotator or automatic detector.

TABLE V
SUMMARY OF THE GROUND TRUTH SURVEY AT PIE RANCH, IN-FLIGHT GEOTAG INTERACTIONS, AND DETECTION RESULTS.

	Count	Ratio
Geotags Placed *(Ground Truth)*	161	—
Geotagged Burrows Viewed	141 of 161	87.6%
Geotagged Burrows Detected	118 of 141	83.7% *(Recall)*
Detection Clusters	318	—
Clusters Matched to Geotags	88 of 251	35.1% *(Precision)*

Human annotation pointclouds are sparse in the sense that they are guided by the inclusion heuristics of Section III-B and will skip borderline examples in favor of labeling consistency. Model detections form denser clouds than the human-placed geotags because they are per-frame projections of the locations of the centroids of noisy detection events, which can include detections of the borderline examples, skipped by humans, based upon the thresholding applied in the post-processing of CNN outputs.

To account for this, we developed a metric which clusters and denoises detection events using DBSCAN, before using a nearest neighbors algorithm to learn whether the clusters of 3D projections of detection events include any of the human-placed geotags, replicating the `eps` inclusion parameter of DBSCAN in a predictive format. See Figure 10 for an illustration of this process. This metric counts true positive detection events while suppressing false positives.

We normalize the output heatmaps by their maximum values, then threshold the raw heatmap into binary detection events at 3% confidence. This low threshold was selected based on empirical tuning to enhance recall while maintaining acceptable precision; the model's activations are typically diffuse and conservative due to the Gaussian hump approach we have taken in labeling. Our DBSCAN denoising filter has the parameter settings of `eps=0.20m` and `min_samples=10`. These settings were empirically found to deliver sensitivity to real targets without oversensitivity to CNN output noise. For our nearest-neighbor prediction stage, we use `eps=0.35m`, which is slightly shorter than length of the 3D 95.4% credible interval (CI) for the worst case of our experiments in Section IV-A (40.4 cm). Under these parameter selections, we derived the detection results presented in Table V.

2) Detection Accuracy: Detection results on the unseen test site are summarized in Table V. Of 161 geotagged burrows, 141 were visible in the aerial imagery. The model successfully detected 118 of these, yielding a recall of 83.7%, indicating a relatively low rate of false negatives. While 251 detection clusters were produced, only 88 matched to ground-truth geotags, corresponding to a precision of 35.1%, indicating an elevated rate of false positives. However, visual inspection revealed that many unmatched clusters actually corresponded to legitimate targets not labeled during ground truth collection. This highlights the difficulty of exhaustive manual annotation in field conditions and degrades the value of false positive counts and the precision metric.

In Figure 11, panels a. and b. show cases of true positive detections while panels c. and d. illustrate the two most common visually-confirmed false positive cases: "edge-of-grass" and "exposed dirt". Additionally, panel a. shows the most common false negative case of target occlusion. Finally, panel b. displays a standout example of the difficulty of defining a false positive case, where the annotator missed two actual positive cases, but the CNN detected them. This is a motivating case for pursuing semisupervised training protocols, so that the model may progressively counteract such false annotations with pseudolabels. We also note that experimenting with different training approaches from the dataset level using *BirdsEye* is relatively easy, due to the system's ability to rapidly re-generate labeled data.

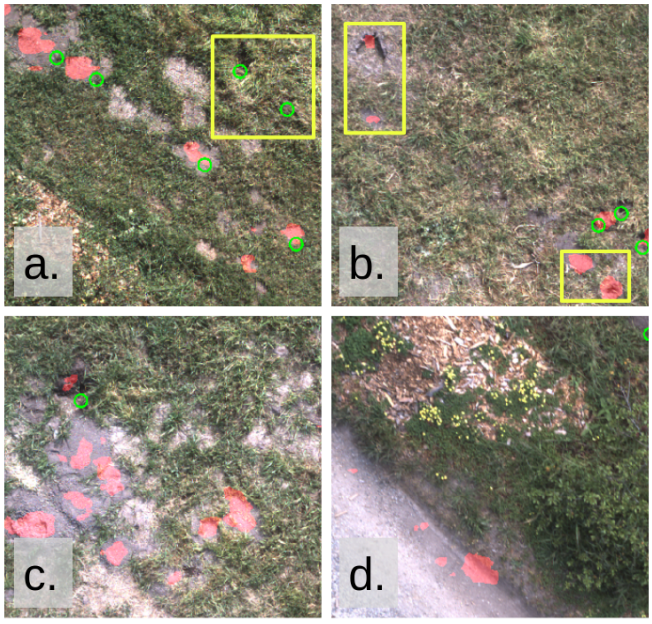


Fig. 11. (a.) A case where the model successfully detects several burrows, but misses two occluded holes. Geotags are represented as green rings and model detections are represented as red patches. (b.) A case where the model successfully detects several burrows which the field annotators failed to mark. (c.) A case where the model segmented damage due to burrows which the annotators skipped. This is adjacent to the common "exposed dirt" failure mode. (d.) An example of the common "edge-of-grass" failure mode, which is a principal contributor to the undeniable False Positive rate.

3) Geospatial Mean Average Precision: To complement the clustering-based analysis, we additionally compute a geospatial mean average precision (g-mAP) metric. The mAP metric [52] is widely used to evaluate object detection models, including in benchmarks like Pascal VOC [53], as it captures both detection accuracy (precision) and localization quality. In our case, we compute g-mAP by matching detections to geotagged annotations within a 0.35 m radius and ranking them by confidence, using a distance threshold in place of the more typical intersection-over-union (IoU) criterion. This adaptation reflects the point-based nature of our geotags, for which IoU is less appropriate.

Without extensive tuning, our baseline CNN achieved a g-mAP of 0.3510 on the single class of the F3 test set. This is comparable to early high-performing competition results on Pascal VOC Challenge [54], demonstrating that the model can reliably detect the target using standard deep learning methods.

While this value is modest compared to typical benchmarks on large, modern competition datasets, it reflects a challenging detection context and provides a strong starting point for further optimization of model architectures, dataset curation methods, and detection postprocessing — all of which is made more accessible by the *BirdsEye* platform.

V. Discussion and Conclusion

This work demonstrates that the BirdsEye system enables accurate, efficient dataset generation for aerial perception tasks in real-world field robotics scenarios. Our geospatial annotation pipeline achieves sub-decimeter 3D georeferencing and sub-30-pixel 2D projection error under both hovering and linear flight, meeting the precision requirements for semantic annotation and object detection in agricultural settings.

The burrow detection case study highlights system generalizability, with 83.7% recall on a held-out site exhibiting heterogeneous terrain and lighting. Although cluster-based precision was measured at 35.1%, qualitative inspection suggests that a significant fraction of false positives correspond to true positives missed by the human-in-the-loop geotagging process. This limitation reflects the inherent subjectivity and sparsity of human annotation when used both for labeling and evaluation.

To address the shortcomings of binary precision-recall metrics under spatial uncertainty, we introduce a geospatial mean average precision (mAP) metric, which accounts for both detection confidence and spatial proximity to ground-truth annotations. Unlike clustering-based evaluation (e.g., DBSCAN), geospatial mAP enables standardized, rank-aware comparisons aligned with broader perception literature.

Quantitative analysis of annotation throughput shows that BirdsEye improves labeling efficiency by a factor of 3–5× relative to GUI-based workflows in similar timeframes, with 22–38× improvements in worker yield when accounting for automation and expert-in-the-loop efficiencies. These gains are critical for sustaining data-centric workflows in environments requiring frequent model retraining.

Several limitations remain. System accuracy degrades during high yaw-rate maneuvers, suggesting the need for either motion filtering in data preprocessing or integration of visual-inertial odometry to enhance pose estimation. Current flight path optimization is limited to simple solutions (e.g., `fast-tsp`), and redundant imagery in the collected datasets points to opportunities for improved flight planning and post hoc frame selection via methods such as keyframe extraction or online hard example mining. While our case study treated burrow signs as a single detection class, future work could explore multiclass annotation pipelines to better capture the visual diversity of pest indicators and improve learning performance; this dataset reconstruction is made practical by *BirdsEye*.

Beyond performance, we emphasize the ethical responsibility of deploying such systems in real-world contexts. We advocate for stakeholder consent and involvement, open-sourced community development, and human-in-the-loop validation as foundational principles for scalable and responsible applications of operational AI.

In closing, we believe BirdsEye provides a practical, scalable solution for generating high-quality training data in field environments. The system integrates calibration, automation, and perception in a modular framework, supporting adaptation to other sensing, mapping, or annotation pipelines. As a deployable, open-source tool built from commodity components, BirdsEye offers a pathway toward reproducible, community-driven dataset creation. Looking forward, we envision that end-to-end open source systems built from off-the-shelf hardware, like *BirdsEye*, can support not only accessible technological advancement and food security, but also equitable and sustainable land stewardship, when deployed with care and in collaboration with local stakeholders.

Acknowledgments

We would like to thank Jake Lee, Liam Asayag, Derick Mathews, Andrew Xu, Andrea Arreortua, and Bryan Suchi for their piloting support, technical insights, and design work on the system's mechanical components. Additionally, this work is supported by the Engineering for Precision Water and Crop Management Program, project award no. 2023-67022-40557, from the U.S. Department of Agriculture's National Institute of Food and Agriculture.

References

[1] Anshuman Bhardwaj, Lydia Sam, F Javier Martín-Torres, Rajesh Kumar, et al., "UAVs as remote sensing platform in glaciology: Present applications and future prospects," *Remote sensing of environment*, vol. 175, pp. 196–204, 2016.

[2] Wouter H. Maes and Kathy Steppe, "Perspectives for Remote Sensing with Unmanned Aerial Vehicles in Precision Agriculture," *Trends Plant Sci.*, vol. 24, no. 2, pp. 152–164, Feb. 2019.

[3] Victor V Klemas, "Coastal and environmental remote sensing from unmanned aerial vehicles: An overview," *Journal of Coastal Research*, vol. 31, no. 5, pp. 1260–1267, 2015.

[4] Krishna Neupane and Fulya Baysal-Gurel, "Automatic Identification and Monitoring of Plant Diseases Using Unmanned Aerial Vehicles: A Review," *Remote Sensing*, vol. 13, no. 19, pp. 3841, Jan. 2021, Number: 19 Publisher: Multidisciplinary Digital Publishing Institute.

[5] Nathalie Pettorelli, Jon Olav Vik, Atle Mysterud, Jean-Michel Gaillard, Compton J. Tucker, and Nils Chr. Stenseth, "Using the satellite-derived NDVI to assess ecological responses to environmental change," *Trends in Ecology & Evolution*, vol. 20, no. 9, pp. 503–510, Sept. 2005.

[6] Telmo Adão, Jonáš Hruška, Luís Pádua, José Bessa, Emanuel Peres, Raul Morais, and Joaquim João Sousa, "Hyperspectral Imaging: A Review on UAV-Based Sensors, Data Processing and Applications for Agriculture and Forestry," *Remote Sensing*, vol. 9, no. 11, pp. 1110, Nov. 2017, Number: 11 Publisher: Multidisciplinary Digital Publishing Institute.

[7] J. L. E. Honrado, D. B. Solpico, C. M. Favila, E. Tongson, G. L. Tangonan, and N. J. C. Libatique, "UAV imaging with low-cost multispectral imaging system for precision agriculture applications," in *2017 IEEE Global Humanitarian Technology Conference*, Oct. 2017, pp. 1–7.

[8] Erika Akemi Saito Moriya, Nilton Nobuhiro Imai, Antonio Maria Garcia Tommaselli, Adilson Berveglieri, Guilherme Henrique Santos, Márcio Augusto Soares, Marcelo Marino, and Thiago Tiedtke Reis, "Detection and mapping of trees infected with citrus gummosis using UAV hyperspectral data," *Computers and Electronics in Agriculture*, vol. 188, pp. 106298, 2021.

[9] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari, "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale," https://storage.googleapis.com/openimages/web/index.html, 2020.

[10] Amazon Web Services, "Registry of open data on aws," https://registry.opendata.aws/, 2025.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[12] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe, "Mots: Multi-object tracking and segmentation," in *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2019, pp. 7942–7951.

[13] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," *arXiv:1603.00831 [cs]*, Mar. 2016, arXiv: 1603.00831.

[14] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé, "CVPR19 tracking and detection challenge: How crowded can it get?," *arXiv:1906.04567 [cs]*, June 2019, arXiv: 1906.04567.

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

[16] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé, "Mot20: A benchmark for multi object tracking in crowded scenes," *arXiv:2003.09003[cs]*, mar 2020, arXiv: 2003.09003.

[17] Andreas Kamilaris and Francesc Xavier Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018.

[18] Sharada Prasanna Mohanty, David P. Hughes, and Marcel Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, pp. 1419, 2016.

[19] Xian Xie, Yujing Ma, Bin Liu, Jian He, Shuang Li, Hui Wang, and Hongyan Wang, "A survey of deep learning techniques for plant disease detection," *Computers and Electronics in Agriculture*, vol. 199, pp. 107126, 2022.

[20] Mingle Xu, Ji-Eun Park, Jaehwan Lee, Jucheng Yang, and Sook Yoon, "Plant disease recognition datasets in the age of deep learning: challenges and opportunities," *Front. Plant Sci.*, vol. 15, pp. 1452551, Sept. 2024.

[21] Damir Yalalov, "AI Model Training Costs Are Expected to Rise from $100 Million to $500 Million by 2030," *Metaverse Post*, Feb. 2023.

[22] Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," 2014.

[23] Michael Buhrmester, Tracy Kwang, and Samuel D. Gosling, "Amazon's Mechanical Turk: A New Source of Inexpensive, Yet High-Quality, Data?," *Perspect. Psychol. Sci.*, vol. 6, no. 1, pp. 3–5, jan 2011.

[24] Inc. Labelbox, "Labelbox: The leading training data platform for ai," 2023, Accessed: Feb 2025.

[25] OpenCV Foundation, "Cvat: Computer vision annotation tool," 2022, Accessed: Feb 2025.

[26] Inc. Roboflow, "Roboflow: End-to-end computer vision dataset management," 2023, Accessed: Feb 2025.

[27] Picterra SA, "Picterra: Ai-powered geospatial image analysis platform," 2021, Accessed: Feb 2025.

[28] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman, "Labelme: A database and web-based tool for image annotation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, 2008.

[29] Liang Yao, Fan Liu, Shengxiang Xu, Chuanyi Zhang, Xing Ma, Jianyu Jiang, Zequan Wang, Shimin Di, and Jun Zhou, "UEMM-Air: A Synthetic Multi-modal Dataset for Unmanned Aerial Vehicle Object Detection," *arXiv*, jun 2024.

[30] Unity Technologies, "Unity Perception package," https://github.com/Unity-Technologies/com.unity.perception, 2020.

[31] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[32] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, pp. 2004–02. IEEE, 2004.

[33] Patryk Cieślak, "Stonefish: An Advanced Open-Source Simulation Tool Designed for Marine Robotics, With a ROS Interface," in *OCEANS 2019 - Marseille*, jun 2019.

[34] Stephan R. Richter, Hassan Abu Alhaija, and Vladlen Koltun, "Enhancing photorealism enhancement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 2, pp. 1700–1715, 2023.

[35] Ahad Rana, "Common crawl – building an open web-scale crawl using hadoop," 2010.

[36] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[37] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A. W. M. van der Laak, Bram van Ginneken, and Clara I. Sánchez, "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42:60-88., Dec. 2017.

[38] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever, "Zero-Shot Text-to-Image Generation," in *International Conference on Machine Learning*, pp. 8821–8831. PMLR, July 2021.

[39] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei, "Language Models are Few-Shot Learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[40] Pengfei Zhu, Longyin Wen, Xiao Bian, Haibin Ling, and Qinghua Hu, "Vision Meets Drones: A Challenge," *arXiv*, apr 2018.

[41] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Wang, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang, "Dota: A large-scale dataset for object detection in aerial images," *arXiv preprint arXiv:1711.10398*, 2017.

[42] TU Graz, "Semantic Drone Dataset," feb 2025, [Online; accessed 13. Feb. 2025].

[43] Xiaoyu Liu, Xugang Lian, Wenfu Yang, Fan Wang, Yu Han, and Yafei Zhang, "Accuracy assessment of a uav direct georeferencing method and impact of the configuration of ground control points," *Drones*, vol. 6, no. 2, pp. 30, 2022.

[44] Darren Turner, Arko Lucieer, and Luke Wallace, "Direct georeferencing of ultrahigh-resolution uav imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 5, pp. 2738–2745, 2013.

[45] Johannes L Schonberger and Jan-Michael Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.

[46] Paul Furgale, Joern Rehder, and Roland Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 03–07. IEEE, 2013.

[47] Adam Korycki, "Tree localization and diameter estimation in coastal redwood forests using neural radiance fields," M.s. thesis, University of California Santa Crus, 2024.

[48] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[49] Nikita Bhatt, Nirav Bhatt, Purvi Prajapati, Vishal Sorathiya, Samah Alshathri, and Walid El-Shafai, "A Data-Centric Approach to improve performance of deep learning models," *Sci. Rep.*, vol. 14, no. 22329, pp. 1–11, sep 2024.

[50] Roger A. Baldwin, Terrell P. Salmon, Robert H. Schmidt, and Robert M. Timm, "Vertebrate pest "research needs" assessment for california agricultural commodities," Tech. Rep., University of California – Kearney Agricultural Research and Extension Center, Parlier, CA, July 2011, Final Report for Vertebrate Pest Control Research Advisory Committee.

[51] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Guide Proceedings*, pp. 226–231. AAAI Press, aug 1996.

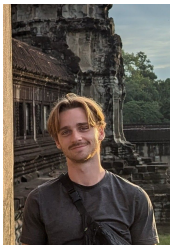
[52] Stephen Robertson, "A new interpretation of average precision," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 689–690.

[53] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *Int. J. Comput. Vision*, vol. 88, no. 2, pp. 303–338, jun 2010.

[54] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman, "The pascal visual object classes challenge: A retrospective," *International journal of computer vision*, vol. 111, pp. 98–136, 2015.

Morgan Masters is a Ph.D. candidate in Electrical and Computer Engineering at UC Santa Cruz and an Agricultural Experiment Station Fellow. His research focuses on automating high-quality dataset generation for AI and data-centric AI, with interests in path planning and next-best view problems in field robotics. He holds B.S. degrees from Iowa State University and an M.S. from the University of Washington.

Adam Korycki is a researcher of Applied Robotics at UC Santa Cruz. He received a B.S. in Computer Engineering and a M.S. in Electrical and Computer Engineering at UC Santa Cruz. His research spans applications of robotics and deep learning in forestry, and soil-powered environmental sensing networks at the far-edge. His research interests are in field robotics, geospatial-AI, and spiking neural networks.



Luca Altaffer holds a B.S. in Applied Physics and an M.S. in Electrical and Computer Engineering from UC Santa Cruz. His work focuses on field robotics, vision-based control systems, and autonomous aerial systems. While at UC Santa Cruz, he conducted research on autonomous data collection for AI model training, with a particular emphasis on applications in agricultural environments.

Nick Kuipers is currently pursuing his M.S in Electrical and Computer Engineering at UC Santa Cruz. His past research involves robotic localization techniques using the Ultra-Wideband (UWB) communications protocol and leveraging Spiking Neural Network (SNN) technology for accurate solar activity prediction in agriculture.

Nikolaas Bender is a research engineer at Charles River Analytics working on practical applications of field robotics. He completed his B.S. at the University of Colorado Boulder, where he was a part of Team MARBLE at the DARPA Subterranean Challenge. He completed his M.S. at the University of California Santa Cruz under the supervision of Dr. McGuire where he specialized in developing robotic agriculture technologies.

Colleen Josephson is an Assistant Professor of Electrical and Computer Engineering and AES Agronomist at UC Santa Cruz. Her research focuses on wireless sensing systems for sustainable practices. She is an associate editor for IEEE Transactions on AgriFood Electronics. Colleen earned her Ph.D. in Electrical Engineering from Stanford and S.B. and M.Eng. degrees from MIT.

Steve McGuire is an Assistant Professor of Electrical and Computer Engineering at UC Santa Cruz. He develops techniques in field robotics to better explore real-world challenging environments. His current projects explore how to leverage advanced robotics in ecology and agriculture to expand scientific inquiry. Steve completed his PhD in Aerospace Engineering Sciences from the University of Colorado Boulder and has been supported by DARPA, ONR, NASA, USDA-NIFA, as well as industry.