

الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية الرياضيات و الإعلام الآلي و الاتصالات السلكية و
اللاسلكية

قسم: الإعلام الآلي



Mémoire de Master en informatique

Spécialité : IAPA

T h è m e



SmartCattleAI : an AI-Powered IOT Platform for intelligent Cattle Monitoring And Farm Automation



▪ Présenté par :

douibat ilyes abdelghani

▪ Dirigé par :

Dr. BENSAID Tayeb

Année universitaire



2025-2026

Acknowledgements

[Thank you god]

Dedications

I closed

Abstract

Dairy cattle monitoring at herd scale presents a persistent challenge: individual animals must be observed continuously, but manual inspection is intermittent and subject to human error. This dissertation presents *Smart-CattleAI*, a multi-modal monitoring system for dairy cattle that integrates computer vision, inertial and positioning sensors, a welfare rule engine, and a natural language agent into a unified web-based platform. Built on the MMCOWS public dataset, the system combines a YOLOv8 object detector with image-based and sensor-based behaviour classifiers operating over a shared seven-class taxonomy, and reconciles their outputs through a late-fusion strategy at the application layer. A configurable threshold-based welfare rule engine translates perception outputs into interpretable alerts for fever, prolonged lying, and low activity, while a pattern-matching natural language agent allows farm managers to query herd state in plain English without external service dependencies. The prototype is implemented as a static-generation pipeline whose offline build produces a self-contained playback artifact for demonstration purposes. Experimental evaluation on a one-hour MMCOWS data slice reveals complementary strengths between vision and sensor modalities, confirms the sensitivity of the alerting layer to threshold configuration, and demonstrates end-to-end coherence through two representative usage scenarios. The work positions multi-modal cattle monitoring as a meaningful case study for integrating perception, decision support, and user interaction in applied agricultural AI.

Keywords: precision livestock farming, dairy cattle monitoring, multi-modal fusion, behaviour classification, object detection, welfare alerting, natural language agent.

Résumé

La surveillance des bovins laitiers à l'échelle du troupeau constitue un défi persistant : chaque animal doit être observé en continu, alors que l'inspection humaine reste intermittente et sujette aux erreurs. Cette dissertation présente *SmartCattleAI*, un système de surveillance multimodal pour bovins laitiers qui intègre la vision par ordinateur, des capteurs inertiels et de positionnement, un moteur de règles de bien-être animal et un agent en langage naturel, au sein d'une plateforme web unifiée. Construit sur le jeu de données public MMCOWS, le système combine un détecteur d'objets YOLOv8 avec des classificateurs de comportement fondés sur l'image et sur le capteur, opérant sur une taxonomie partagée de sept classes, et réconcilie leurs sorties par une stratégie de fusion tardive au niveau de la couche applicative. Un moteur de règles de bien-être à seuils configurables traduit les sorties de perception en alertes interprétables pour la fièvre, le décubitus prolongé et la faible activité, tandis qu'un agent de correspondance de motifs permet aux éleveurs d'interroger l'état du troupeau en langage naturel sans dépendance à des services externes. Le prototype est mis en œuvre sous la forme d'un pipeline de génération statique dont la construction hors ligne produit un artefact de lecture autonome à des fins de démonstration. L'évaluation expérimentale sur une tranche d'une heure du jeu MMCOWS met en évidence les forces complémentaires des modalités vision et capteur, confirme la sensibilité de la couche d'alerte à la configuration des seuils, et démontre la cohérence de bout en bout à travers deux scénarios d'utilisation représentatifs. Ce travail positionne la surveillance multimodale des bovins comme une étude de cas représentative pour l'intégration de la perception, de l'aide à la décision et de l'interaction

utilisateur dans l'intelligence artificielle agricole appliquée.

Mots-clés : élevage de précision, surveillance des bovins laitiers, fusion multimodale, classification comportementale, détection d'objets, alertes de bien-être, agent en langage naturel.

ملخص

تُمثّل مراقبة الأبقار الحلوب على مستوى القطيع تحدياً مستمراً: إذ يجب مراقبة كل حيوان على حدة بصفة مستمرة، في حين يظل الفحص البشري متقطعاً وعرضةً للخطأ. تقدّم هذه الأطروحة نظام SmartCattleAI وهو نظام مراقبة متعدد الوسائط لأبقار الحليب، يدمج رؤية الحاسوب وأجهزة استشعار القصور الذاتي والتحديد الجغرافي الداخلي، إلى جانب محرك قواعد رفاهية الحيوان ووكيل لغة طبيعية، ضمن منصة ويب موحّدة. بُني النظام على مجموعة البيانات العامة MMCOWS، ويجمع بين كاشف الكائنات YOLOv8 ومصنّفات سلوكية مستندة إلى الصورة وإلى المستشعر، تعمل وفق صنف مشترك من سبع فئات، ووفقاً بين مخرجاتها عبر استراتيجية الدمج المتأخر على مستوى طبقة التطبيق. يُترجم محرك قواعد رفاهية الحيوان القائم على عتبات قابلة للضبط مخرجات الإدراك إلى تنبيهات قابلة للتفسير تتعلق بالحمى والرقود المطول وانخفاض النشاط، في حين يُتيح وكيل مطابقة الأنماط للمزارعين الاستعلام عن حالة القطيع بلغة طبيعية دون الحاجة إلى خدمات خارجية. نُفذ النموذج الأولي على شكل خط أنابيب توليد ثابت ينتج أثراً جاهزاً للتشغيل لأغراض العرض. أظهر التقييم التجريبي على شريحة زمنية مدتها ساعة واحدة من مجموعة MMCOWS نقاط قوة تكاملية بين وسائط الرؤية والمستشعر، وأكّد حساسية طبقة التنبيه لإعدادات العتبات، وبرهن على الاتساق الشامل من خلال سيناريوهي استخدام تمثيليّين. يُرسّخ هذا العمل مراقبة الأبقار متعددة الوسائط بوصفها دراسة حالة نموذجية لدمج الإدراك ودعم القرار والتفاعل مع المستخدم في الذكاء الاصطناعي الزراعي التطبيقي.

الوسائط، متعدد الدمج الحلوب، الأبقار مراقبة الدقيقة، الماشية تربية **الكلمات المفتاحية:** الطبيعية. اللغة وكيل الرفاهية، تنبيهات الكائنات، كشف السلوك، تصنيف

Contents

Acknowledgements	1
Dedications	2
Abstract	3
Résumé	4
ملخص	6
Introduction	15
1 Problem statement	15
2 Motivations	15
3 Objectives	16
4 Thesis structure	17
1 Background and Literature Review	18
1.1 Precision Livestock Farming	18
1.1.1 Definition of precision livestock farming	18
1.1.2 Modalities and indicators	19
1.1.3 Continuous monitoring in dairy systems	21
1.2 Computer Vision for Livestock Monitoring	22
1.2.1 Definition of vision-based monitoring	22
1.2.2 Object detection as a perception task	23
1.2.3 Challenges of computer vision in dairy environments	24
1.2.4 Importance of behavior recognition in cattle vision	25
1.3 Sensor-Based Monitoring	25

1.3.1	Inertial sensing	25
1.3.2	Localization with UWB and RFID	26
1.3.3	Thermal and physiological sensing	27
1.4	Multi-Modal Fusion	28
1.4.1	Why fuse modalities	28
1.4.2	Early, late, and hybrid fusion	29
1.5	Decision Support and Welfare Alerting	31
1.5.1	Rule-based welfare engines	31
1.5.2	Agent-based and conversational interfaces	31
1.6	Dairy Cattle Welfare Context	32
1.6.1	Overview of the dairy production setting	32
1.6.2	Welfare indicators	32
1.6.3	Industry challenges	33
1.7	The MMCOWS Dataset	34
1.7.1	Dataset overview	34
1.7.2	Annotation methodology	34
1.8	Behavior Label Taxonomy	35
1.8.1	Overview of behavior categories	35
1.8.2	Taxonomy in context	36
1.9	State of the Art and Related Work	36
1.9.1	Related work	38
1.10	Applications of AI in Livestock Farming	39
1.10.1	Health and welfare monitoring	39
1.10.2	Reproduction management	40
1.10.3	Resource use and barn design	40
1.10.4	Dataset construction and research support	41
1.11	Conclusion	41
2	Methodology	42
2.1	Introduction	42
2.2	Dataset Configuration and Preparation	42
2.2.1	Data slice selection and acquisition	42
2.2.2	Multi-camera configuration	43

2.2.3	Sensor instrumentation	44
2.2.4	Cow identity management	45
2.2.5	Dataset structure and organization	45
2.2.6	Data preparation pipeline	46
2.3	System Architecture Overview	48
2.3.1	Overall architecture	48
2.3.2	Build pipeline	49
2.3.3	Playback model	50
2.4	Detection Module	51
2.4.1	YOLOv8 architecture	51
2.4.2	MMCOWS fold-1 weights	51
2.4.3	Normalized bounding-box output	52
2.5	Image-Based Behavior Classifier	52
2.5.1	Cropped cow patches as input	52
2.5.2	Class mapping and checkpoint	53
2.6	Sensor-Based Behavior Classifier	53
2.6.1	Inertial windowing	53
2.6.2	Sensor classifier checkpoint	54
2.7	Multi-Modal Reconciliation	54
2.7.1	Three parallel behavior streams	54
2.7.2	Confidence handling and tie-breaking	54
2.8	Welfare Rule Engine	55
2.8.1	Fever thresholds	55
2.8.2	Prolonged lying detection	56
2.8.3	Low-activity detection	56
2.9	Natural Language Agent	57
2.9.1	Pattern-matching design	57
2.9.2	Intent taxonomy	58
2.10	Dashboard Design	58
2.10.1	Page organization	58
2.10.2	Multi-camera grid and pen map	59
2.10.3	Cow cards and metrics panel	60
2.10.4	Threshold settings with live preview	60

2.11	Conclusion	61
3	Experiment and Results	62
3.1	Introduction	62
3.2	Evaluation Setup	62
3.3	Detection Performance	63
3.3.1	Per-camera observations	63
3.3.2	Latency considerations	63
3.4	Behavior Classification Results	64
3.4.1	Image classifier observations	64
3.4.2	Sensor classifier observations	64
3.4.3	Agreement with ground truth	65
3.5	Welfare Alert Evaluation	66
3.5.1	Threshold sensitivity	66
3.5.2	False-positive analysis	67
3.6	Discussion of Results	67
3.7	Conclusion	68
4	System Implementation and Prototype	69
4.1	Introduction	69
4.2	Technology Stack	69
4.2.1	Frontend stack	69
4.2.2	Backend stack	70
4.2.3	Build pipeline implementation	70
4.3	Prototype Dashboard Overview	71
4.4	Scenario 1 – Normal Herd Monitoring	72
4.5	Scenario 2 – Alert/Anomaly Detection	74
4.5.1	Settings and live-preview threshold tuning	74
4.5.2	AI Agent investigation	75
4.6	Conclusion	76
	Conclusion and Future Work	78

List of Figures

1.1	Main observation modalities in precision livestock farming.	20
1.2	Common challenges in precision livestock farming.	22
1.3	Illustration of a vision-based livestock monitoring pipeline.	23
1.4	A real ankle-accelerometer window from the MMCOWS slice (cow T03, sampled at 10 Hz). The shaded region marks one 100-sample (~ 10 s) window of the kind fed to the sensor-based classifier.	26
1.5	UWB occupancy density of the ten instrumented cows over the one-hour slice, overlaid on the barn pen boundary. Hotter cells mark zones where the herd spends more time. . .	27
1.6	Core body temperature traces for the ten instrumented cows across the slice, with the 39.0°C warning and 39.5°C critical thresholds. No animal crosses the warning line in this window.	28
1.7	Early, late, and hybrid multi-modal fusion strategies. . . .	30
1.8	Behavioral and physiological indicators used in dairy welfare monitoring.	33
1.9	Ground-truth behaviour distribution over the slice (cow-seconds across all sixteen cows). Lying and standing dominate, while walking, drinking, and licking are rare.	36
2.1	Four synchronised camera views from the MMCOWS slice at a single timestamp, covering the feed bunk, lying area, walking lane, and a wide barn view.	44

2.2	Multi-modal data preparation pipeline used by SmartCattleAI: raw data families, dataset loaders, the per-tick build loop with its alignment tolerances, and the static artifacts produced.	47
2.3	Overall architecture of SmartCattleAI, with offline build and online playback layers.	49
2.4	Runtime data flow: the offline build emits <code>timeline.json</code> and the frame tree, which the playback hook loads and feeds to the dashboard components.	50
2.5	The YOLOv8 detection module: backbone, multi-scale neck, decoupled head, and post-processing into normalised detections.	51
2.6	A single cropped cow patch of the kind fed to the image-based behaviour classifier, taken from a ground-truth bounding box in the slice.	53
2.7	Multi-modal reconciliation of image, sensor, and ground-truth behavior streams.	55
2.8	The welfare rule engine maps monitored inputs onto threshold rules and human-readable alerts, summarised into a per-cow severity.	56
2.9	The natural language agent normalises the question, matches it against intent patterns, queries the in-memory herd state, and returns a templated answer — entirely on the client.	57
2.10	Frontend architecture: an application shell hosting four pages, with the Overview page composing the main monitoring components.	59
2.11	Top-down pen map: each instrumented cow is placed at its real UWB position and coloured by its current behaviour.	60
3.1	Illustrative agreement between image, sensor, and ground-truth behavior streams.	65

3.2	Welfare-alert sensitivity computed on the real slice: number of cows flagged as a function of the fever-warning threshold (left) and the prolonged-lying threshold (right), with the defaults marked.	66
4.1	The Reports page, showing report cards with summary metrics and the daily / weekly / monthly / custom report templates.	71
4.2	Overview page combining multi-camera grid, pen map, metrics panel, and cow roster under normal herd conditions. .	73
4.3	Threshold settings with live preview: lowering the prolonged-lying threshold immediately updates per-row alert counts on the Overview page.	75
4.4	The AI Agent page: a live herd snapshot above a local chat interface answering natural-language questions about individual cows.	76

List of Tables

1.1	Examples of precision livestock farming modalities and indicators	20
1.2	Representative detection architectures used in livestock vision	24
1.3	SmartCattleAI behavior taxonomy and welfare relevance .	35
1.4	Representative AI-based livestock monitoring studies reported in the literature	37
1.5	Example of behavioral signals used in dairy reproduction management	40
2.1	Top-level directory layout of the MMCOWS data slice used by SmartCattleAI	46
2.2	Default thresholds used by the SmartCattleAI welfare rule engine	57
2.3	Representative intents supported by the SmartCattleAI natural language agent	58
3.1	Illustrative per-camera detection observations on the MMCOWS data slice	63
3.2	Illustrative per-class behavior of the image classifier on MMCOWS	64
3.3	Illustrative alert volume under different threshold configurations	67
4.1	Main components of the SmartCattleAI technology stack .	70

Introduction

1 Problem statement

Precision livestock farming is built around a straightforward idea: continuous, individual, data-driven observation of farm animals throughout their productive lives [1]. As cheap sensors, embedded vision systems, and machine learning have spread through agriculture, the field has had to develop pipelines that turn raw observations into actionable signals about animal health, behaviour, and welfare [2]. Dairy cattle are the natural focus: they combine high economic value with strict welfare expectations, and their behavioural patterns are detailed enough that intermittent human observation routinely misses early signs of disease, discomfort, or productivity loss [1]. Modern herd sizes make this harder.

Despite the availability of multiple sensing technologies — cameras, accelerometers, ultra-wideband tags, and physiological sensors — existing monitoring solutions rarely integrate them into a coherent, user-facing platform. Most address a single modality, leaving farm managers without a unified view of herd state. The central problem addressed in this dissertation is: *how can heterogeneous data streams from a dairy barn be integrated into a single, interpretable monitoring system that supports real-time welfare decisions at the individual animal level?*

2 Motivations

Behaviour recognition and welfare alerting sit near the centre of this work because they make each animal’s interaction with its environment measur-

able over time [9, 8]. Behavioural monitoring captures resting patterns, feeding activity, locomotion, and abnormal events, and feeds higher-level tasks such as lameness detection, oestrus monitoring, calving prediction, and disease screening [13]. It also draws on image-based, sensor-based, and multi-modal methods, since all three families have been applied to cattle observation [10, 16]. Multi-modal cattle monitoring is therefore the framing used throughout this dissertation.

The motivation for building *SmartCattleAI* rests on three observations. First, the MMCOWS dataset [17] now provides a public, synchronised multi-modal benchmark for dairy cattle, enabling reproducible end-to-end research. Second, the gap between individual perception components — detectors, classifiers, rule engines — and a usable farm tool remains wide; bridging it requires system-level design as much as modelling skill. Third, interpretability is as important as accuracy in farm settings: alerts that farmers cannot explain or tune are alerts they will ignore.

3 Objectives

The central goal of this dissertation is to study AI-based livestock monitoring through the design and implementation of *SmartCattleAI* — a multi-modal system for dairy cattle that combines computer vision, inertial and positioning sensors, a welfare rule engine, and a natural language agent [2, 16]. The work explains the role of multi-modal observation in precision livestock farming, lays out the theoretical foundations of object detection and behaviour recognition, and examines how heterogeneous data sources can be integrated into a single coherent view of the herd [4, 6, 9]. Throughout, dairy cattle monitoring serves as a concrete test case for evaluating end-to-end intelligent agricultural systems [1].

A second objective is to make the dairy setting’s specific difficulties explicit: occlusion, lighting variation, individual identity, and behavioural context all shape system performance in ways worth understanding [3, 8]. To that end, the dissertation examines several component families — YOLO-family detectors, image-based and sensor-based behaviour classi-

fiers, threshold-based welfare rule engines, and a local conversational interface for end users [6, 9, 10]. The aim is not only to describe individual algorithms but to show how their integration produces a working monitoring platform, and to position multi-modal cattle monitoring as a meaningful case study for applied agricultural AI.

4 Thesis structure

This dissertation is organised into four chapters, each addressing a distinct phase of the work, framed by this Introduction and a closing Conclusion and Future Work section.

Chapter 1 covers the conceptual background: precision livestock farming, computer-vision and sensor-based monitoring, multi-modal fusion, and the use of welfare rule engines in dairy contexts [1, 2, 16]. It also presents the dairy cattle welfare setting, introduces the MMCOWS dataset and its behaviour taxonomy, and reviews representative studies from the literature [8, 9, 10, 17].

Chapter 2 sets out the methodological framework. It begins with the dataset configuration and data preparation pipeline that produce the structured timeline consumed by the system, then presents the SmartCattleAI architecture in detail — the YOLOv8 detection module, image-based and sensor-based behaviour classifiers, multi-modal reconciliation, welfare rule engine, natural language agent, and dashboard design [17, 6, 12].

Chapter 3 presents the experimental evaluation: per-camera detection observations, per-class behaviour-classification analysis, welfare-alert sensitivity under different threshold configurations, and a discussion of the results in relation to the broader literature [8, 9, 12].

Chapter 4 details the system implementation and prototype, describing the technology stack and build pipeline before demonstrating the end-to-end platform through two representative usage scenarios — normal herd monitoring and alert/anomaly detection [2, 16].

Chapter 1

Background and Literature Review

Precision livestock farming combines sensing, computation, and decision support to make individual animal management measurable and data-driven [1, 2]. This chapter provides the conceptual foundations on which SmartCattleAI is built. It begins with an overview of precision livestock farming and its modalities, then examines computer-vision and sensor-based monitoring as the two main perception families, before turning to multi-modal fusion and welfare decision support [16, 12]. The chapter then introduces the dairy cattle welfare context and the MMCOWS dataset that provides the empirical foundation for this work, presents the behaviour taxonomy used throughout the system, reviews representative studies from the literature, and concludes with a survey of AI applications in livestock farming [17, 9].

1.1 Precision Livestock Farming

1.1.1 Definition of precision livestock farming

Precision livestock farming applies sensing, modelling, and control technologies to the continuous management of individual animals on commercial farms [1, 2]. The goal is to track each animal in real time, flag de-

viations from normal behaviour, and support farm decisions that improve productivity, welfare, and resource use. What used to be a herd-level operation becomes an information-driven activity in which each animal is treated as a measurable, interpretable entity [2].

Herd sizes have grown faster than human capacity to observe individual animals with adequate attention [2]. Automated sensing systems fill that gap, capturing short-lived events, gradual trends, and rare anomalies that would otherwise go unnoticed. They also support record-keeping, traceability, and the longitudinal datasets that research and benchmarking depend on [17].

1.1.2 Modalities and indicators

Precision livestock farming draws on a wide range of sensing modalities, each capturing a different facet of animal state [2, 9]. Fixed and on-animal cameras supply visual, video, and thermal signals [3, 8]. Accelerometers and gyroscopes mounted on collars, halters, or ankles record posture and movement [10]. Ultra-wideband, RFID, and GPS tags log where each animal spends its time [11]. Sensors for core body temperature, rumen activity, and milk yield add direct physiological evidence [13]. No single channel covers everything [16].

Table 1.1 sketches how the field operates across these observation levels, from raw visual input to direct physiological measurement.

Table 1.1: Examples of precision livestock farming modalities and indicators

Modality	Sensor type	Indicator captured	Example application
Vision	Fixed RGB camera	Posture, location in scene, group behavior	Detecting feeding or lying cows
Vision	Thermal camera	Surface temperature, fever screening	Detecting elevated body temperature
Inertial	Ankle / neck accelerometer	Activity, gait, posture transitions	Lameness and oestrus detection
Positioning	UWB tag in barn	Location, residence time per zone	Feed bunk and lying area usage
Physiological	Rumen bolus / CBT sensor	Body temperature, rumen pH	Disease and calving alerts
Production	Milking robot logs	Yield per visit, conductivity	Mastitis screening

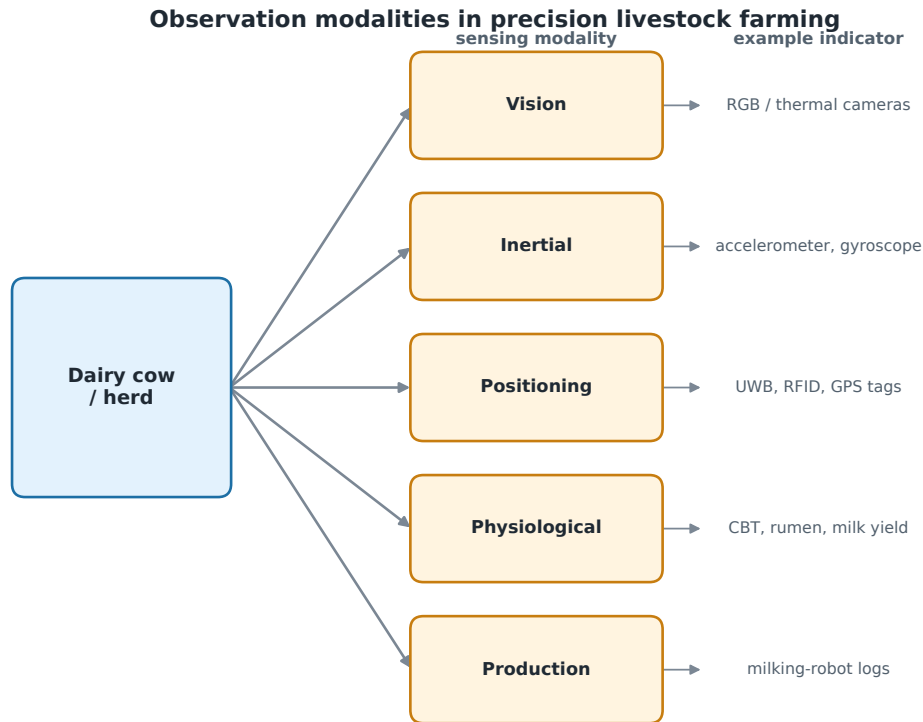


Figure 1.1: Main observation modalities in precision livestock farming.

Each modality suits different purposes. Vision handles group behaviour and spatial occupation; accelerometers drive activity-based welfare scoring; positioning systems show how each animal uses the barn’s resources [3, 9,

11]. Physiological sensors add direct health signals — particularly useful for fever, calving, and rumination monitoring [13]. In practice, several are combined, and the quality of any one channel can quietly limit every downstream stage [16]. Figure 1.1 maps each modality onto the indicator it captures.

1.1.3 Continuous monitoring in dairy systems

Continuous monitoring gives herd managers and computational models the structured observations they need to spot regularities and flag deviations [2, 13]. When measurements are tied to individual cows over time, models can learn a baseline and detect when an animal drifts from it — particularly useful for early disease detection, where a single isolated measurement rarely tells the full story [9, 10, 12].

Continuously annotated datasets carry the same advantage into research: they let different systems be compared under shared conditions and recurring error sources be analysed [17]. This dissertation uses the MMCOWS dataset and the SmartCattleAI prototype as the experimental basis because they are both theoretically grounded and methodologically manageable — a concrete example of how multi-channel observation makes different perception approaches comparable [16].

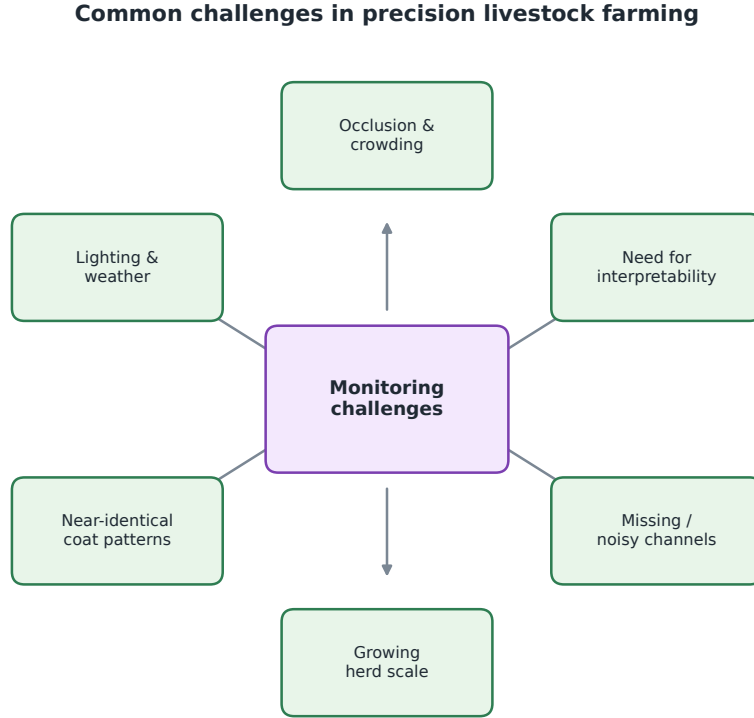


Figure 1.2: Common challenges in precision livestock farming.

The recurring difficulties of the field are summarised in Figure 1.2: occlusion and crowding, lighting and weather variation, near-identical coat patterns, growing herd scale, missing or noisy channels, and the constant need for interpretable outputs.

1.2 Computer Vision for Livestock Monitoring

1.2.1 Definition of vision-based monitoring

Vision-based livestock monitoring is the task of analysing images or video from animal environments to extract structured information: identity, location, posture, behaviour [3, 8]. Cameras are cheap, non-invasive, and capture group-level context that contact sensors cannot, which explains their wide adoption in agricultural AI [2]. Interpreting each frame requires situating it within a wider scene, making the task harder than plain image

classification [7]. In most practical pipelines, vision sits as an intermediate layer between the camera and any deeper herd analysis.

Behaviour recognition, body condition scoring, lameness detection, and individual identification all depend on detection and tracking outputs as preliminary input [14]. Even in fully multi-modal models, visual information helps with context and interpretation [16, 17]. Vision-based monitoring is therefore both a practical farm tool and a standard reference point for comparing perception approaches.

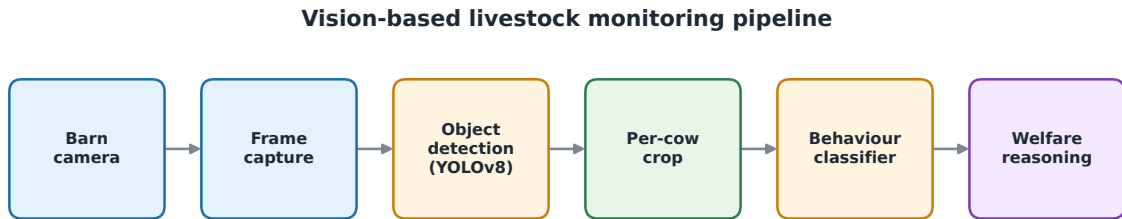


Figure 1.3: Illustration of a vision-based livestock monitoring pipeline.

Figure 1.3 shows the typical chain: a barn camera supplies frames, an object detector localises each animal, per-cow crops feed a behaviour classifier, and the resulting labels drive welfare reasoning.

1.2.2 Object detection as a perception task

Object detection is usually framed as a combined localisation and classification problem: the input is an image and the output is a set of bounding boxes, each tagged with a class label and a confidence score [4, 5]. For a frame I , the model predicts a list of triples $(b_1, c_1, s_1), (b_2, c_2, s_2), \dots, (b_n, c_n, s_n)$, where b_i is a bounding box, c_i a class, and s_i a confidence score. Farm scenes are inherently multi-instance — several animals typically visible at once — so livestock detection fits the multi-object localisation mould far better than global image classification [8, 3].

Table 1.2 groups detection architectures into families and shows the methodological options available for livestock vision.

Table 1.2: Representative detection architectures used in livestock vision

Family	Representative model	Main characteristic
Two-stage	Faster R-CNN	Region proposal followed by classification; high accuracy, slower inference
Single-shot	SSD	Multi-scale anchor boxes predicted in a single pass
Single-shot	YOLOv3 / v4	Grid-based prediction, real-time speed, widely adapted to agriculture
Anchor-free	YOLOv8	Decoupled head, anchor-free design, strong default in modern livestock work

Consider a fixed barn camera watching several cows: each animal must be localised, possibly identified, and associated with a behaviour label [8, 3]. Getting this right requires the model to combine local appearance with broader scene context [5]. In dairy environments, occlusion, near-identical coat patterns, and shifting lighting make the task considerably harder than detection on curated photographic datasets. Detection pipelines address the problem by exploiting multi-scale features, anchor structures, and attention mechanisms to identify several animals per frame [4, 6, 18].

1.2.3 Challenges of computer vision in dairy environments

Dairy barns are visually complex, which makes livestock monitoring harder than detection in most other domains [3, 8]. A single camera typically captures several animals with overlapping bodies, partial occlusion by infrastructure, and varying distances from the lens — all of which complicate localisation and identity assignment. Many animals also look nearly identical: Holstein patterns differ only in detail, so the system must lean on context, position, and temporal continuity to maintain identity across frames.

Two cows lying close together in a free-stall area can collapse into a single detection on a poorly tuned model — one reason dairy vision requires evaluation under realistic barn conditions rather than controlled benchmarks. Indoor and outdoor environments, day and night cycles, and

seasonal lighting changes all vary in contrast, colour balance, and shadowing, complicating the design of unified vision systems [2]. Commercial deployments add further variation: camera placement, cleanliness, and network bandwidth differ between farms. Robust dairy vision models need to handle this variation rather than simply performing well on one site.

1.2.4 Importance of behavior recognition in cattle vision

Behaviour recognition is central to cattle vision because so much welfare-relevant information is encoded in the postural and temporal structure of activity [8, 9]. Walking, standing, lying, feeding, drinking, and licking are strong indicators of comfort, health, and production state [13]. Accurate detection without behavioural classification still misses most of what matters. In dairy systems, turning visual observations into behavioural labels is the primary goal.

Traditional systems combined handcrafted geometric and motion features with classical classifiers [3, 12]. Neural approaches learn the relevant patterns from data, using convolutional backbones, temporal pooling, and contextual features [7]. Either way, behaviour recognition dominates both performance and error analysis.

1.3 Sensor-Based Monitoring

1.3.1 Inertial sensing

Inertial sensing uses accelerometers and gyroscopes mounted on collars, halters, ear tags, or ankles to record an animal’s motion and posture [9, 10]. The sensors produce time series capturing the intensity, frequency, and orientation of movement, from which behaviours such as walking, standing, lying, and ruminating can be separated. In dairy cattle, the approach has been applied to lameness, oestrus, calving, and abnormal activity levels [13, 14, 15]. Its practical appeal is straightforward: low cost, low bandwidth,

and a direct tie to the individual animal.

In practice, raw inertial signals are summarised over short sliding windows to extract descriptive features — mean, variance, dominant frequency, signal-magnitude area — which then feed classifiers that emit behaviour labels at the window level. Modern systems also pass raw or lightly processed windows directly to neural models that learn their own representations. The approach works with both classical and deep learning pipelines.

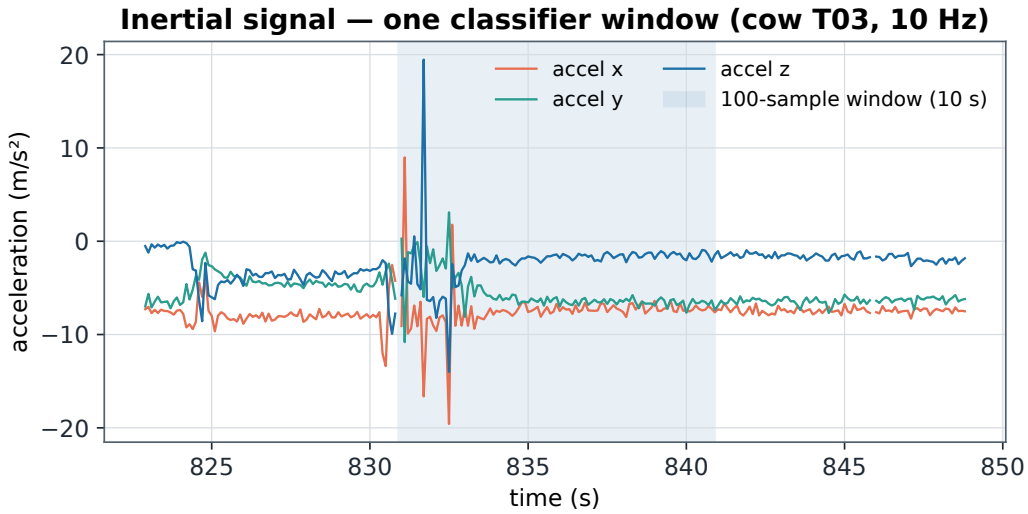


Figure 1.4: A real ankle-accelerometer window from the MMCOWS slice (cow T03, sampled at 10 Hz). The shaded region marks one 100-sample (~ 10 s) window of the kind fed to the sensor-based classifier.

Figure 1.4 shows one such window taken from the dataset used in this work: the three acceleration axes carry a clear motion burst that a classifier can map to an active behaviour.

1.3.2 Localization with UWB and RFID

Indoor localisation of livestock is typically handled by ultra-wideband or RFID systems, which deliver sub-metre position estimates under realistic barn conditions [11]. UWB uses time-of-flight measurements between animal-worn tags and fixed anchors, making it robust to multipath and partial occlusion. The resulting trajectories show how each animal uses

the barn’s resources — feed bunk, lying area, water trough, milking robot [2] — and add spatial context that no single camera can supply [16].

Localisation data also aggregates into welfare indicators: residence time per zone, distance walked per day, social proximity between animals. These metrics catch welfare issues that show up first as spatial changes — reduced feed-bunk attendance, unusual isolation from the herd [13]. The system therefore serves both real-time and longitudinal roles.

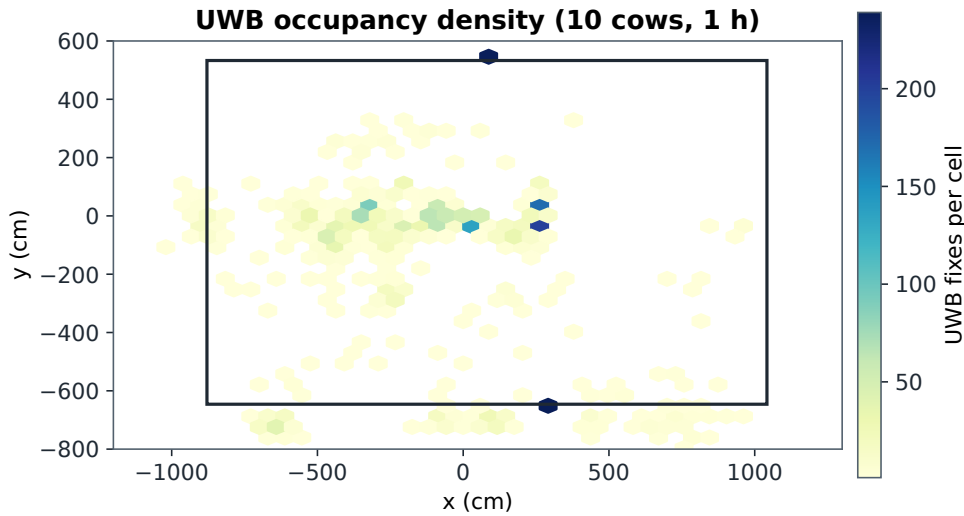


Figure 1.5: UWB occupancy density of the ten instrumented cows over the one-hour slice, overlaid on the barn pen boundary. Hotter cells mark zones where the herd spends more time.

Figure 1.5 aggregates one hour of UWB fixes from the dataset into an occupancy heatmap, making resource-zone usage directly visible.

1.3.3 Thermal and physiological sensing

Thermal and physiological sensors capture direct health signals — core body temperature, surface skin temperature, rumen pH, milk conductivity [2, 13]. Core body temperature is particularly useful: elevated values often precede clinical signs of disease by hours or days. These measurements may come from rumen boluses, intra-vaginal sensors, or thermal cameras targeting specific body regions. Pairing them with behavioural observations gives the monitoring pipeline both indirect and direct evidence of health

[16].

Physiological signals are most informative for fever detection, calving prediction, and metabolic transitions around parturition. Because they reflect internal state directly, they often supply stronger evidence than behaviour alone — especially when the behavioural picture is ambiguous. The trade-off is cost and invasiveness: physiological sensors are more burdensome than vision or inertial systems, which limits their deployment at scale. This tension motivates the multi-modal designs discussed later.

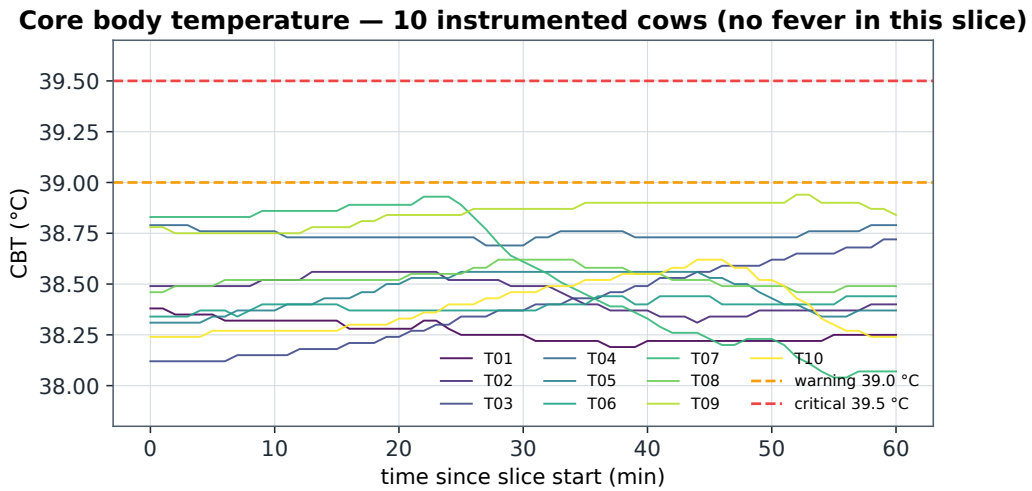


Figure 1.6: Core body temperature traces for the ten instrumented cows across the slice, with the 39.0°C warning and 39.5°C critical thresholds. No animal crosses the warning line in this window.

The real core-body-temperature traces in Figure 1.6 stay comfortably below the warning threshold, which is itself a useful observation: the slice contains no febrile animal, so fever alerts should be silent under default thresholds.

1.4 Multi-Modal Fusion

1.4.1 Why fuse modalities

Multi-modal fusion combines several observation channels to produce a more reliable view of the monitored system than any single channel alone

[16]. In dairy monitoring, vision describes the global scene and group behaviour, inertial sensors capture each animal’s motion in detail, and positioning systems record where each animal spends its time [11, 9]. Each modality has failure modes the others can compensate for — a property that matters most for welfare decisions, where both false alarms and missed events carry real cost [17, 2].

Fusion also helps with data availability. Cameras fail because of dust or network problems; sensors fall off or run out of battery; individual animals leave the field of view of any single camera [3]. A multi-modal system keeps producing useful information when one channel degrades.

1.4.2 Early, late, and hybrid fusion

Fusion architectures are usually grouped into three strategies: early, late, and hybrid [16]. Early fusion concatenates raw features from several modalities and passes them to a single model that learns joint representations. Late fusion processes each modality with its own model and combines only the resulting predictions or scores at the end. Hybrid fusion mixes the two — for example by extracting modality-specific features early and integrating them at one or more intermediate layers. Each strategy makes different trade-offs in robustness, modularity, and the handling of missing data [17].

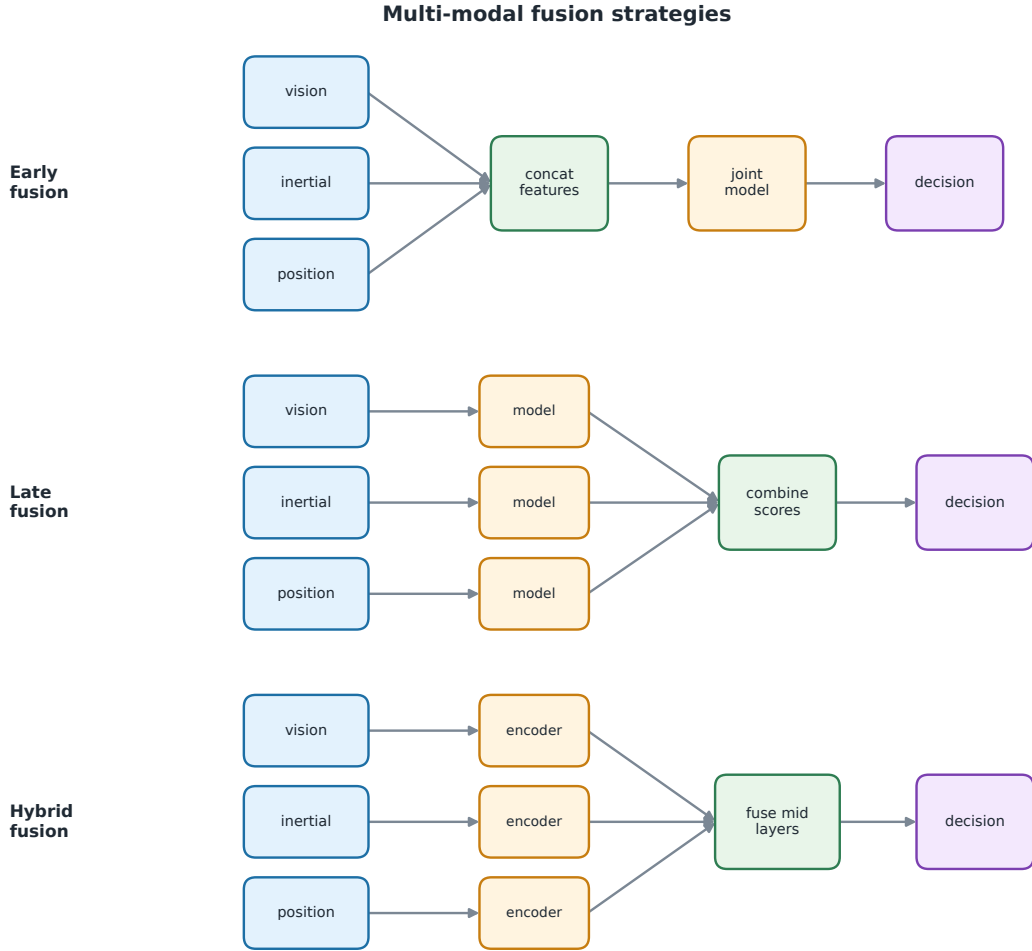


Figure 1.7: Early, late, and hybrid multi-modal fusion strategies.

As Figure 1.7 illustrates, the three strategies differ mainly in where the modality streams meet. In practice, late fusion is the easiest to deploy: each modality can be developed, trained, and replaced independently. This matters in agriculture, where sensors are added and removed over time as hardware changes [2, 11]. Early fusion can produce stronger joint representations but degrades more when channels are missing or noisy. Smart-CattleAI takes a late-fusion-leaning approach, exposing per-modality predictions and reconciling them at the application layer.

1.5 Decision Support and Welfare Alerting

1.5.1 Rule-based welfare engines

Rule-based welfare engines apply expert-defined thresholds to monitored variables to generate human-readable alerts [12, 2]. A system may flag a cow if her core body temperature exceeds 39.0°C, or if she has been lying continuously for more than sixty minutes [13]. Such rules are transparent, easy to explain to farmers, and easy to audit when an alert is disputed. They also encode existing veterinary knowledge directly into the system.

The weaknesses appear when behaviour is ambiguous, contextual, or individually variable. Fixed thresholds can fire false alarms for animals whose baseline differs from the herd average, and miss subtle deviations that never cross an explicit limit. Modern systems tend to combine rule-based alerts with learning-based components that adjust thresholds or rank alerts by severity. SmartCattleAI keeps rule-based alerts as the primary user-facing mechanism for interpretability, while making thresholds configurable so farm-specific variation can be accommodated.

1.5.2 Agent-based and conversational interfaces

Recent work in precision livestock farming has explored conversational interfaces that let farmers query the monitoring system in natural language [2]. A focused question — “which cows have fever?”, “where is Daisy?” — replaces manual dashboard navigation, lowering the cognitive cost of accessing complex multi-modal data. These interfaces also surface long-tail information that rarely makes it onto the main screens of a monitoring dashboard.

Large language models are an obvious candidate, but simpler pattern-based agents are often enough when the underlying state is structured and the query space well bounded. Pattern-based agents are easier to deploy on-premises, require no external services, and behave predictably in safety-relevant contexts. SmartCattleAI adopts a pattern-matching natural-language layer for these reasons, leaving the door open for an LLM-

backed variant in future work.

1.6 Dairy Cattle Welfare Context

1.6.1 Overview of the dairy production setting

Dairy cattle are among the most heavily monitored farm animals because they combine high economic value with strict welfare expectations and complex behavioural patterns [2, 13]. Computationally, they are particularly interesting: daily activities differ across posture, motion, and location, which makes the production system an informative target for multi-channel observation [8, 9]. The setting exposes both the strengths and the limits of perception and decision methods under real conditions.

Unlike monitoring problems with a simple behavioural structure, dairy operations produce continuous, individual data covering several activity categories over each twenty-four-hour cycle. Each animal alternates between feeding, lying, walking, standing, drinking, and social interaction, and the meaning of any single observation often depends on its position in the daily activity budget. Dairy data is information-rich, but also computationally demanding.

1.6.2 Welfare indicators

Several welfare indicators are routinely used to assess individual dairy cows in modern monitoring systems [2, 13]. The standard list includes lying time, lying bout length, walking activity, feeding time, water intake, ruminating activity, and physiological measurements such as core body temperature [12]. Each indicator captures a different facet of welfare, and combinations tend to yield more reliable assessments than any single measure [16].

This has direct consequences for system design. A monitoring system that relies on a single indicator can misread welfare state when it ignores contextual signals and concurrent measurements. Modern dairy

monitoring platforms therefore tend to favour architectures that preserve per-modality detail, or models that can reason jointly about several indicators.

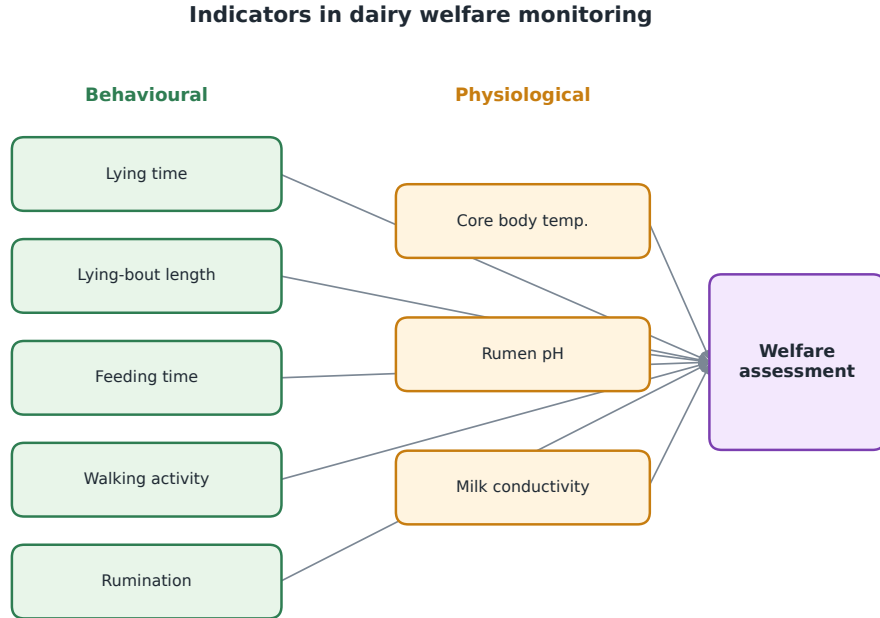


Figure 1.8: Behavioral and physiological indicators used in dairy welfare monitoring.

Figure 1.8 groups the standard indicators into behavioural and physiological families, both of which feed a combined welfare assessment.

1.6.3 Industry challenges

Dairy monitoring is also shaped by the operational realities of commercial farms [2, 3]. Cameras and sensors must function reliably in dusty, humid, high-traffic environments with minimal manual maintenance. Network bandwidth inside a barn is typically limited, which caps how much raw data can travel to off-site servers. These constraints favour edge-friendly, lightweight pipelines — and motivate the static-generation strategy in research prototypes like SmartCattleAI.

A second concern is interpretability [12]. Farmers and veterinarians rely on monitoring outputs to make practical decisions, and they need to understand why a given alert fired. Black-box models that output risk scores without explanation are hard to trust in the field, especially when treatment or culling decisions are involved. That is why SmartCattleAI keeps rule-based thresholds, configurable parameters, and visible per-modality outputs at the centre of its design.

1.7 The MMCOWS Dataset

1.7.1 Dataset overview

The MMCOWS dataset is a public multi-modal resource for dairy cattle monitoring that combines synchronised cameras with on-animal sensors and expert annotations [17]. It is one of the first public benchmarks where vision, inertial, and positional data are released together with consistent ground truth [16]. The combination makes it possible to develop and evaluate detection, behaviour classification, and multi-modal fusion components within a single coherent setup. The dataset is central to this dissertation — both as the source of training and evaluation data and as the empirical foundation of the SmartCattleAI prototype.

In practical terms, MMCOWS covers a working dairy environment in which several cows are observed simultaneously by multiple cameras, while a subset of the herd wears inertial, positioning, and physiological sensors. The dataset is organised so that visual, inertial, positional, and physiological observations can be aligned in time and associated with the same individual animal.

1.7.2 Annotation methodology

Ground-truth behaviour labels in MMCOWS are produced by trained annotators who watch synchronised video and assign behaviour categories to each animal at fixed time intervals [17, 12]. The annotation guidelines

fix a taxonomy of behaviours, define exclusion rules for ambiguous events, and set tie-breaking conventions when multiple behaviours overlap. The resulting labels are stored as per-cow time series and serve as the reference against which automatic classifiers are evaluated [9].

Alongside behaviour labels, MMCOWS provides bounding-box annotations for the cows visible in each camera frame. Boxes are stored in the standard YOLO text format, with one line per object encoding the cow identifier, centre coordinates, width, and height in normalised image coordinates [4]. The format is widely supported by modern detection frameworks and matches YOLOv8’s input expectations [6].

1.8 Behavior Label Taxonomy

1.8.1 Overview of behavior categories

SmartCattleAI uses a seven-class behaviour taxonomy derived from the MMCOWS annotation scheme [17, 12]. The classes are *walking*, *standing*, *feeding (head up)*, *feeding (head down)*, *licking*, *drinking*, and *lying*, covering the main behaviours observed in a free-stall dairy environment [9]. The taxonomy is rich enough to support meaningful welfare reasoning yet small enough to keep classifier outputs interpretable for end users [2].

Table 1.3: SmartCattleAI behavior taxonomy and welfare relevance

Class	Type	Welfare relevance
Walking	Locomotion	Activity level, gait, lameness screening
Standing	Posture	Idle time at the feed bunk or alley
Feeding (head up)	Feeding	Time at the feed bunk in chewing posture
Feeding (head down)	Feeding	Time at the feed bunk in eating posture
Licking	Maintenance	Grooming activity, comfort indicator
Drinking	Maintenance	Water intake proxy
Lying	Rest	Rest time, lying bout length, comfort

Table 1.3 summarises the behaviours and the welfare aspect each one informs.

1.8.2 Taxonomy in context

The seven behaviour classes appear in three different streams in SmartCattleAI: as ground-truth labels from MMCOWS, as outputs of the image-based behaviour classifier, and as outputs of the sensor-based classifier [17, 16]. Sharing the same label space across all three streams is what makes multi-modal reconciliation meaningful — the system can compare and combine predictions that live in the same semantic vocabulary.

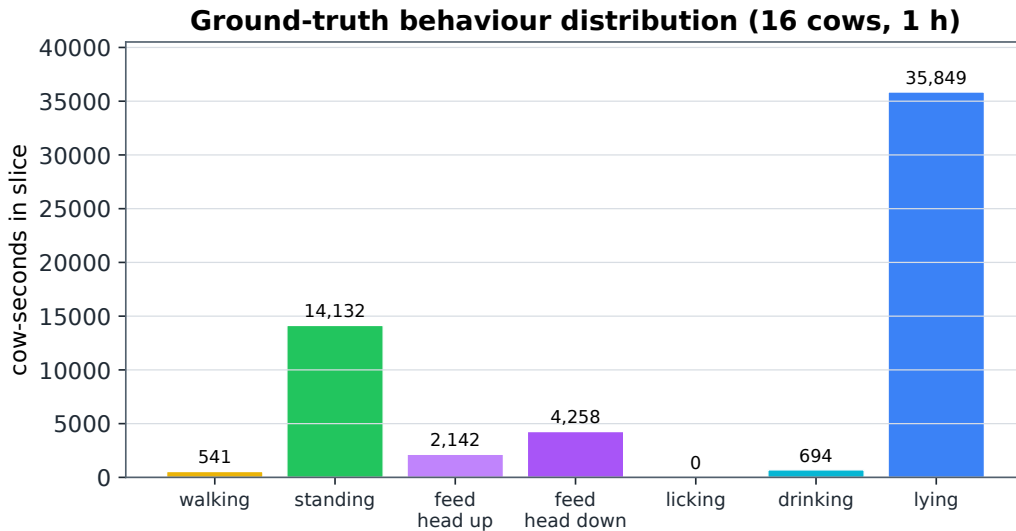


Figure 1.9: Ground-truth behaviour distribution over the slice (cow-seconds across all sixteen cows). Lying and standing dominate, while walking, drinking, and licking are rare.

The class distribution in Figure 1.9, computed directly from the ground-truth labels, exposes a strong imbalance: lying and standing account for the vast majority of cow-seconds, whereas drinking and walking are rare and licking is absent in this window. This imbalance is important context for the per-class results reported in Chapter 3.

1.9 State of the Art and Related Work

Table 1.4 lists representative studies with their approach, setting, and main result [3, 12, 13, 11, 8, 9, 10, 17]. The trend is clear: rule-based and feature-engineered models have given way to neural and multi-modal approaches.

Table 1.4: Representative AI-based livestock monitoring studies reported in the literature

Paper	Approach	Dataset / Setting	Main idea	Result
Nasirahmadi et al. (2017)	Computer vision	Indoor cattle and pigs	Reviews machine vision methods for behavior detection in livestock	Vision is mature for posture and group behavior
Arcidiacono et al. (2017)	Threshold rules	Dairy cow accelerometer	Real-time classifier with rule-based thresholds for behavior recognition	Strong agreement with expert observation
Borchers et al. (2017)	Machine learning	Activity, lying, rumination	Predicts calving from sensor-derived behavioral features	Reliable short-horizon calving alerts
Vázquez Diosdado et al. (2020)	UWB localization	Indoor dairy farm	UWB-based individual position tracking and resource-use analysis	Sub-meter accuracy under barn conditions
Wang et al. (2021)	Deep learning vision	Cattle video	End-to-end CNN-based cattle behavior recognition	Outperforms handcrafted feature baselines
Riaboff et al. (2022)	Sensor ML	Accelerometer studies	Systematic review of accelerometer-based livestock behavior prediction	Highlights need for shared datasets
Pavlovic et al. (2021)	Sensor ML	Neck-collar accelerometer	Multi-class behavior classification with classical and neural models	Comparable performance for tested classes
MMCOWS (2024)	Multi-modal	Dairy barn (vision + sensors)	Public multi-modal cattle monitoring dataset with annotations	Enables reproducible multi-modal research

The main open questions concern dataset availability, behavioural coverage, and model generalisation.

1.9.1 Related work

Several studies have approached cattle monitoring from different angles — vision-based behaviour recognition, sensor-based activity classification, and multi-modal dataset construction [3, 8, 9, 10, 11, 17]. Nasirahmadi et al. (2017) reviewed machine-vision approaches for monitoring cattle and pigs, focusing on posture and group behaviour [3]. The review showed that vision-based methods apply across several behaviour categories despite considerable barn variation, and it flagged the recurring difficulty of crowded, occluded, visually homogeneous scenes.

Arcidiacono et al. (2017) proposed a real-time threshold-based classifier for dairy cattle behaviour recognition [12]. Their results showed that simple rules, tuned with expert input, achieve strong agreement with human observation — useful evidence that interpretable thresholds hold up against more complex approaches. The study is a useful reference for any work that combines machine learning with explicit thresholds [2, 13].

Borchers et al. (2017) tackled calving prediction using machine-learning models trained on activity, lying, and rumination features extracted from sensors [13]. Ensembles and boosted trees produced reliable short-horizon predictions, particularly when several behavioural channels were combined [16]. The study illustrates how directly continuous behavioural observation feeds time-critical farm decisions.

Vázquez Diosdado et al. (2020) showed that ultra-wideband positioning systems can be deployed in indoor dairy farms with sub-metre accuracy [11]. The work emphasised spatial context — how each cow uses the barn’s resource zones — rather than restricting evaluation to activity-only behaviours [2]. UWB is what makes those resource-zone interactions measurable.

Wang et al. (2021) investigated cattle behaviour recognition with end-to-end convolutional architectures trained on video [8]. The work confirmed

that deep learning consistently outperforms handcrafted vision features on realistic barn data [7], and flagged a recurring caveat: strong benchmark performance does not guarantee strong on-farm performance.

More recently, the MMCOWS dataset [17] introduced a public multi-modal benchmark combining synchronised cameras with on-animal sensors and behaviour annotations. It supports reproducible research on detection, behaviour classification, and multi-modal fusion within a single coherent setup [16], and provides the empirical foundation on which SmartCattleAI is built.

Taken together, the reviewed studies trace a clear arc from feature-based and rule-based approaches to deep neural architectures and multi-modal datasets. They also expose recurring challenges: occlusion, behavioural variability, and the need for explainable alerts. These observations motivate treating multi-modal cattle monitoring as a representative case study for integrating computer vision, sensors, and decision support in agricultural AI.

1.10 Applications of AI in Livestock Farming

1.10.1 Health and welfare monitoring

AI-based livestock monitoring feeds directly into health and welfare management through continuous, individual observation [2, 13]. A continuous behavioural and physiological record lets downstream stages detect deviations, anomalies, and gradual trends that hint at emerging issues — often before any clinical sign appears. Behavioural and physiological data are typically the first signals that prompt decisions about treatment, separation, or veterinary inspection.

The contribution extends beyond acute disease detection. Welfare assessment depends on distinguishing resting, feeding, social, and locomotor behaviours, since these categories organise the cow’s daily activity budget. A more accurate behavioural profile improves any welfare score or audit

built on top of it — including those driven by regulatory compliance rather than clinical concern.

1.10.2 Reproduction management

In reproduction management, continuous monitoring reduces uncertainty around oestrus, mating, and calving [13, 2]. These events are short-lived and easily missed when human observation is intermittent, and economic outcomes often depend on catching them at the right time. Behavioural indicators tighten the link between observation and intervention.

Table 1.5: Example of behavioral signals used in dairy reproduction management

Event	Indicator type	Behavioral signature	Management decision
Oestrus	Activity	Sharp increase in steps and mounts	Insemination timing
Calving	Lying / activity	Restlessness followed by extended lying	Calving pen preparation
Post-partum	Feeding / rumination	Reduced feeding time and rumination	Metabolic check

Table 1.5 maps behavioural signals onto reproductive events and the management decisions they inform.

Reproductive performance directly affects herd productivity, and explicit monitoring exposes information that downstream modules would otherwise miss. Even in barns that rely heavily on production sensors, behavioural monitoring adds value for analysis, evaluation, and decision support.

1.10.3 Resource use and barn design

AI-based monitoring tracks which animals occupy which zones and for how long, supporting resource-use analysis [11, 2]. The same data exposes competition at the feed bunk, overcrowding in lying areas, and underuse of facilities such as brushes or water troughs. Scaled up to barn design,

the occupancy record can inform decisions about layout — both for daily operations and longer-term investments.

1.10.4 Dataset construction and research support

AI-based monitoring also enables richly annotated multi-modal datasets that can be reused in research, benchmarking, and system development [17, 9]. Annotated datasets support both model training and evaluation, and document the properties of a particular farm, breed, or production stage. Annotation support tools blend human expertise with automatic predictions to speed up labelling, and this interaction is most useful when high-quality resources are needed and fully manual annotation would be too slow.

1.11 Conclusion

This chapter introduced precision livestock farming and placed dairy cattle at its centre as both a high-value monitoring target and a representative case study for applied agricultural AI [1, 2]. It framed vision-based and sensor-based monitoring as detection and time-series classification problems, examined the specific difficulties of dairy observation, and reviewed representative studies from the literature [4, 9, 3, 8, 13, 11, 10, 17]. It also introduced the dairy cattle welfare context, the MMCOWS dataset, the seven-class behaviour taxonomy, and multi-modal fusion and rule-based welfare engines as core building blocks of modern monitoring systems [16, 12].

The next chapter turns to the methodological framework — dataset configuration, data preparation, and the full design of the SmartCattleAI system.

Chapter 2

Methodology

2.1 Introduction

This chapter sets out the complete methodological framework adopted for SmartCattleAI. It begins with the dataset configuration and data preparation steps that convert the raw MMCOWS slice into the structured timeline the system consumes, then explains how each system component is designed and how the resulting predictions are combined into a coherent view of herd state [17, 2, 16]. The methodology rests on a modular system design: each modality is processed by its own component, and the resulting predictions are exposed to the application layer in a uniform way [16]. The design is organised around three directions — a perception layer combining object detection and behaviour classification, a reasoning layer built on a welfare rule engine, and an interaction layer providing both a dashboard and a natural language agent [12].

2.2 Dataset Configuration and Preparation

2.2.1 Data slice selection and acquisition

The full MMCOWS dataset is large, and not all of it is needed for a research prototype. SmartCattleAI therefore works with a focused data slice covering a short but representative time window [17]. The slice is chosen

to include diverse behaviours, several instrumented animals, and at least one welfare-relevant event — a prolonged lying bout, or an elevated body temperature [12, 2]. The result is rich enough to demonstrate the full pipeline while staying small enough to ship as a static artifact served from the frontend.

A dedicated script handles acquisition of the data slice: it downloads the relevant portion of MMCOWS and extracts it into the expected directory layout [17]. The script encapsulates assumptions about file naming, archive structure, and timestamp conventions, making it possible to refresh the slice reproducibly. The acquisition tooling also performs minimal sanity checks — the presence of expected camera folders, the availability of behaviour CSVs for each cow — to catch partial downloads early.

2.2.2 Multi-camera configuration

SmartCattleAI uses four cameras to observe the barn from different viewpoints, matching the multi-view nature of MMCOWS [17, 3]. The cameras are positioned so that the lying area, feeding area, and main walking lanes are covered with overlapping fields of view. The overlap increases the probability that each cow remains visible in at least one camera at any given time [8], and it lets the system maintain individual identity across viewpoints — essential for per-animal reasoning.

MMCOWS slice — four synchronised camera views @ t=1690299476

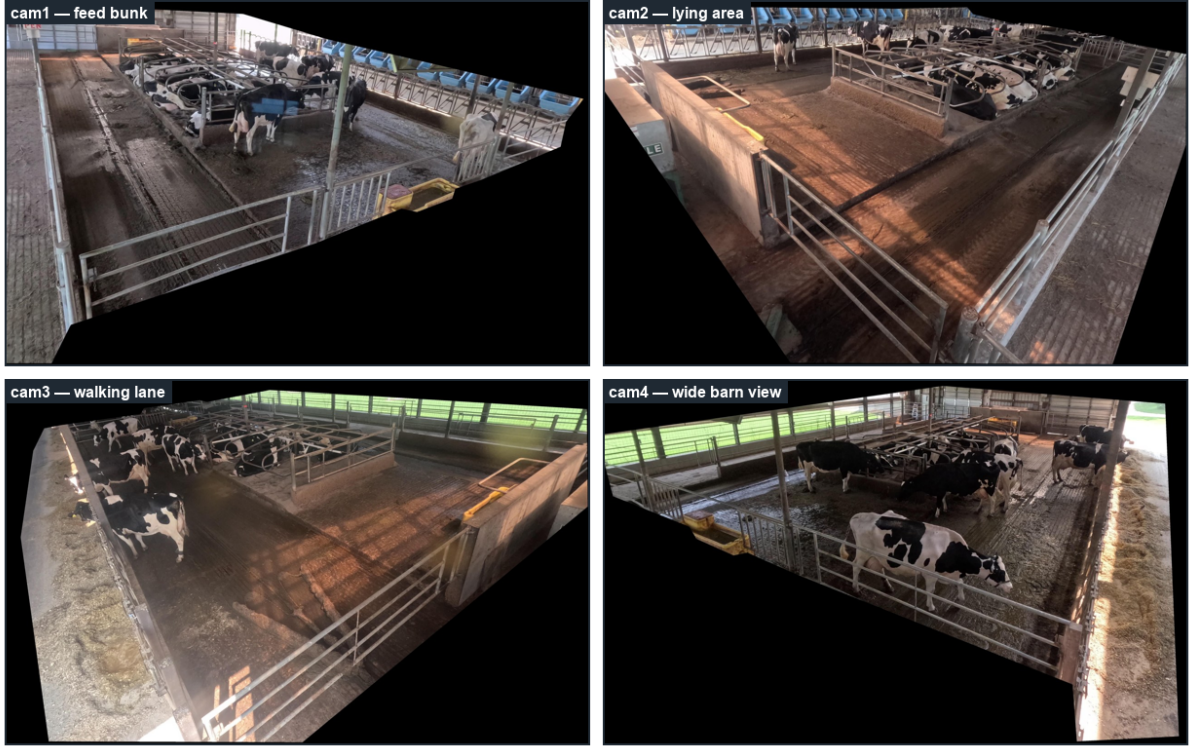


Figure 2.1: Four synchronised camera views from the MMCOWS slice at a single timestamp, covering the feed bunk, lying area, walking lane, and a wide barn view.

Figure 2.1 shows the four views at one synchronised instant. Each camera produces a sequence of JPEG frames at a fixed sampling rate, indexed by Unix timestamps in the build pipeline. The frames live in a structured directory layout under `cam1`, `cam2`, `cam3`, and `cam4`, which makes nearest-neighbour lookup by timestamp straightforward.

2.2.3 Sensor instrumentation

A subset of the herd wears on-animal sensors that record motion, position, and temperature alongside the camera feeds [17, 9, 11]. In the SmartCattleAI configuration, ten animals (T01–T10) are equipped with ankle-mounted inertial measurement units, ultra-wideband tags for indoor localisation, and optionally core body temperature sensors [2]. The remaining cows are observed only through the cameras — the realistic case where not every animal in a herd is instrumented.

The sensor channels are deliberately complementary [16]. The IMU stream records high-frequency acceleration and magnetometer readings and captures fine-grained motion patterns [10]. The UWB stream records position estimates in barn coordinates, supporting occupancy and trajectory analysis. The CBT stream records core body temperature, a direct health signal [13]. Together, they make MMCOWS a representative multi-modal benchmark for dairy monitoring research.

2.2.4 Cow identity management

SmartCattleAI manages a herd of sixteen cows identified as C01 through C16, each associated with a canonical name, a unique tag identifier, and the Holstein breed [17]. The canonical names — Daisy, Bella, Molly, Luna, and so on — let the user-facing components, including the natural language agent, refer to animals the way farmers actually talk about their herd [2]. Internally, the system relies on stable numeric identifiers, keeping display strings decoupled from computational state.

Only the first ten cows (T01–T10) wear the full sensor set; the remaining six are observed exclusively through the cameras [17]. The asymmetry is deliberate: it reflects the realistic case where sensor coverage is partial, and it forces the system to handle missing modalities gracefully [16].

2.2.5 Dataset structure and organization

The dataset slice used by SmartCattleAI follows a fixed directory layout that separates raw images, sensor recordings, and labels into distinct top-level folders [17]. Images are organised by camera and timestamp; sensors by modality and animal; labels by frame timestamp for bounding boxes and by animal for behaviours.

Table 2.1 summarises the top-level layout. The separation lets each modality be loaded, inspected, and replaced independently [16], and it mirrors the modular design of the backend, where each loader handles a single data type.

Table 2.1: Top-level directory layout of the MMCOWS data slice used by SmartCattleAI

Folder	Subfolders	Content
images/	cam1, cam2, cam3, cam4	Per-camera JPEG frames named by Unix timestamp
sensors/	imu/, uwb/, cbt/	Per-modality CSV files, one file per instrumented cow
labels/	bboxes/, behaviors/	YOLO-format bounding boxes per frame and behavior CSVs per cow

2.2.6 Data preparation pipeline

The data preparation pipeline is the bridge between the raw MMCOWS slice and the single static artifact the frontend consumes. It is implemented as one offline build script (`build_demo.py`) that composes three dataset loaders, walks the slice in time order, and emits one tick of timeline data per step [17, 16]. Figure 2.2 gives the end-to-end view: the three raw data families on the left are read by dedicated loaders, combined inside a per-tick build loop, and written out as `timeline.json` plus a tree of camera frames.

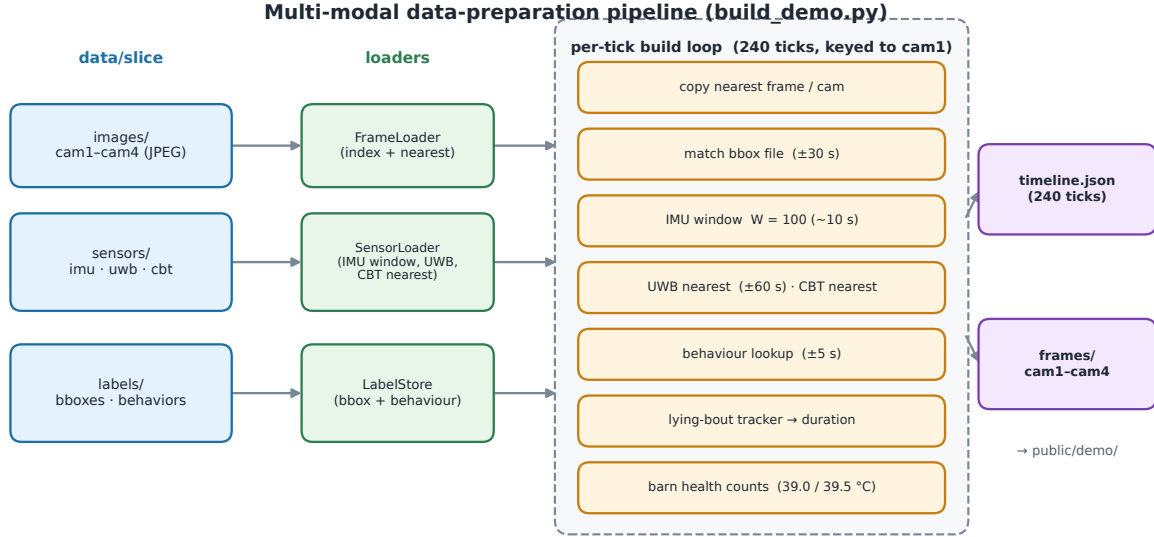


Figure 2.2: Multi-modal data preparation pipeline used by SmartCattleAI: raw data families, dataset loaders, the per-tick build loop with its alignment tolerances, and the static artifacts produced.

Frame indexing by timestamp. The first stage indexes camera frames by their Unix timestamps [17]. The `FrameLoader` walks each of the four camera folders, parses the integer timestamp from each JPEG filename, and stores a per-camera sorted list of timestamps together with a timestamp-to-path map. Because the list is sorted, the nearest frame to any requested time is found with a binary search ($O(\log n)$). The build does not invent its own time grid: it ticks on the *actual* frame timestamps of the primary camera that fall inside the configured slice window, guaranteeing that the camera-grid image and the YOLO bounding-box file for a tick share exactly the same timestamp.

Sensor synchronisation. The three sensor streams are sampled very differently: the IMU runs at roughly 10 Hz, UWB positions arrive about every fifteen seconds, and core body temperature is logged about once a

minute [16]. For each instrumented cow and each tick, the `SensorLoader` extracts the trailing inertial window of $W = 100$ samples (~ 10 s); if fewer than one hundred samples precede the tick the window is dropped rather than padded [9, 10]. The UWB and CBT streams are aligned by nearest neighbour, each guarded by a maximum tolerated gap (`MAX_UWB_GAP_S` = 60 s for UWB) so that stale readings are never associated with a tick [11].

Label parsing and normalisation. Bounding-box lines are parsed into normalised coordinate tuples and grouped by frame timestamp; the box file nearest the frame is accepted only within ± 30 s [4, 6]. Behaviour rows are parsed into per-cow time series; the label nearest a tick is accepted only within ± 5 s [12, 2]. Integer behaviour codes are mapped through the canonical `BEHAVIOR_GT` table to the seven-class taxonomy, and timestamps are converted to a single Unix time base.

Static timeline construction. For each of the 240 ticks the build loop emits a record with four fields: the tick timestamp, a `barn` summary, a list of per-cow records, and the per-camera detections. A stateful per-cow lying-bout tracker carries a `lying_since` timestamp across ticks, used to derive `lying_duration_sec`. The whole structure is serialised into a single minified `timeline.json` (about 2.9 MB for the 240-tick slice), and the nearest real frame for each camera is copied into `public/demo/frames/cam{1--4}`.

Handling missing data. The dataset is only partially instrumented, and even instrumented animals have gaps [16]. Each alignment rule is therefore allowed to fail gracefully: an out-of-tolerance lookup or an under-full IMU window yields an empty field rather than dropping the entire tick or animal. This preserves the structure of the timeline even when individual modalities are incomplete.

2.3 System Architecture Overview

2.3.1 Overall architecture

SmartCattleAI is organised as a static-generation pipeline: an offline build process produces self-contained artifacts, and the browser plays them back

[2, 16]. The build runs the Python backend on the MMCOWS data slice and emits a JSON timeline together with a tree of camera frames, all written to the frontend’s public directory [17]. The frontend, implemented in Next.js, loads these artifacts and replays them as if they were live observations of the herd.

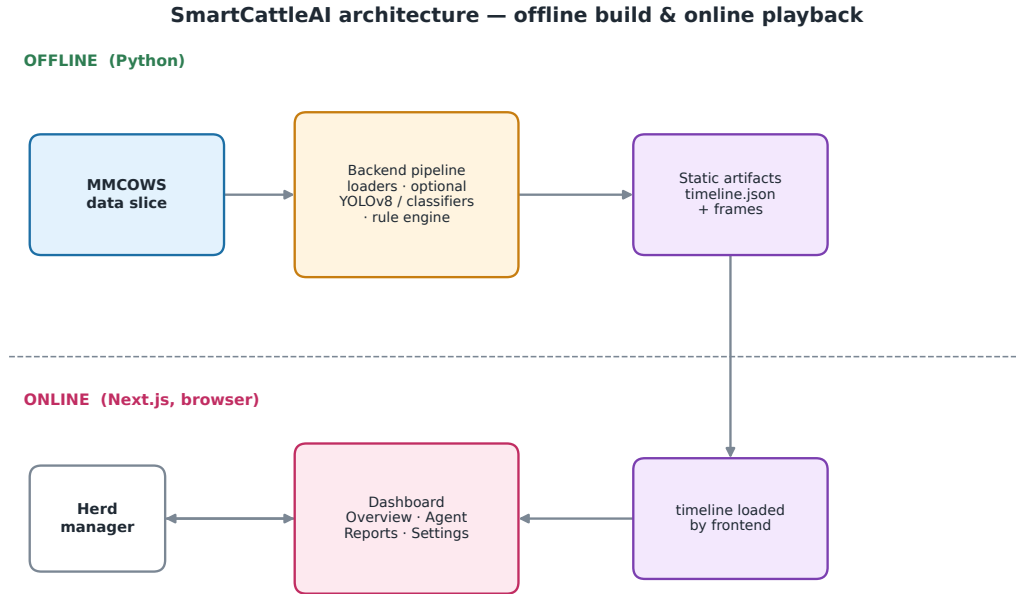


Figure 2.3: Overall architecture of SmartCattleAI, with offline build and online playback layers.

This architecture, sketched in Figure 2.3, suits research prototypes because it minimises the deployment surface and produces deterministic demonstrations. The same backend components can be reused in a live FastAPI deployment, so the static-generation mode is best understood as a particular configuration rather than a structural limit.

2.3.2 Build pipeline

The build pipeline consumes the MMCOWS data slice and produces the static artifacts the frontend needs [17, 16]. Internally, it composes the dataset loaders, the optional perception models, and the welfare rule engine

into a single sequential process that emits one tick of timeline data at a time [12].

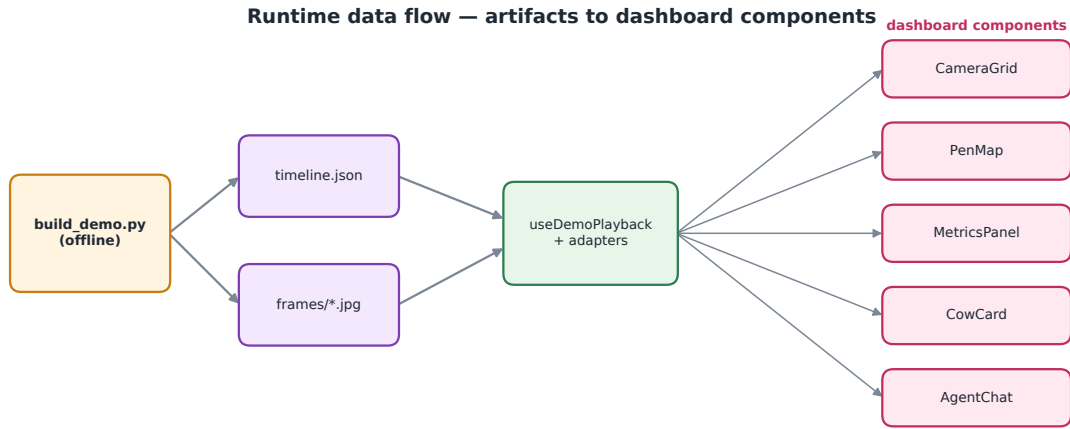


Figure 2.4: Runtime data flow: the offline build emits `timeline.json` and the frame tree, which the playback hook loads and feeds to the dashboard components.

The pipeline is deterministic for the same input slice and configuration. Figure 2.4 traces how the resulting artifacts move at runtime, from the build script through the playback hook to the individual dashboard components.

2.3.3 Playback model

The frontend loads the static timeline and plays it back at a user-controlled speed [2]. The default is one tick per second of wall-clock time. The user can pause, resume, and seek the playback — especially useful when investigating a specific event such as an alert or a behaviour transition. This client-side playback model keeps the dashboard responsive even on modest hardware, since the heavy perception work has already been done offline.

2.4 Detection Module

2.4.1 YOLOv8 architecture

The detection module is built around YOLOv8 — a modern single-shot object detector with an anchor-free decoupled head and a backbone-neck-head design [6, 5, 4]. YOLOv8 suits livestock detection well: it offers a favourable balance between accuracy and inference latency, and has a mature open-source toolchain [8]. In SmartCattleAI it localises cows in each camera frame and produces normalised bounding boxes for downstream components.

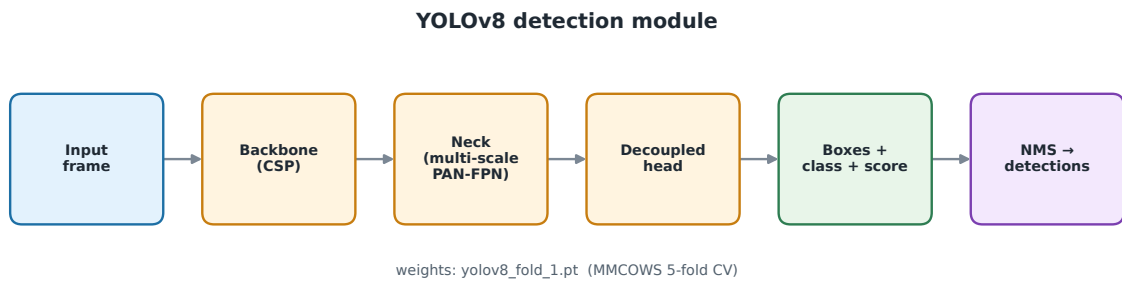


Figure 2.5: The YOLOv8 detection module: backbone, multi-scale neck, decoupled head, and post-processing into normalised detections.

The architecture, summarised in Figure 2.5, follows the standard pattern: a convolutional backbone for feature extraction, a multi-scale neck that aggregates features at different resolutions, and a detection head that predicts bounding boxes and class scores at multiple scales [5]. SmartCattleAI treats YOLOv8 as a black-box component whose weights are loaded from a pre-trained checkpoint.

2.4.2 MMCOWS fold-1 weights

For livestock-specific detection, SmartCattleAI uses YOLOv8 weights trained on the MMCOWS dataset under a five-fold cross-validation scheme [17, 6]. The default checkpoint corresponds to fold one and is stored as `yolov8_fold_1.pt` in the prototype’s `pretrained/` directory. A livestock-specific checkpoint

matters here because generic detectors trained on broad image collections tend to underperform on tightly framed barn scenes with near-identical coat patterns [8, 3].

By default the build pipeline relies on the ground-truth bounding boxes provided by MMCOWS. The YOLOv8 detector can be enabled explicitly when the corresponding model file is available — it is fully optional from the build pipeline’s point of view.

2.4.3 Normalized bounding-box output

The detection module emits a list of normalised bounding boxes for each camera frame, with each box encoding the cow identifier, the centre coordinates, and the width and height in image-relative units [4, 6]. Normalised coordinates are resolution-independent and easy to render as overlays on top of the displayed frames in the browser. These boxes are stored per tick in the static timeline and consumed by the frontend at display time.

2.5 Image-Based Behavior Classifier

2.5.1 Cropped cow patches as input

The image-based behaviour classifier takes cropped patches of individual cows as input, rather than full camera frames [8, 3, 7]. The crops come either from the detection module’s bounding boxes or from the MMCOWS ground truth, strongly localising the input to a single animal [17]. This framing lets the classifier focus on the posture and motion of a single subject and reduces background interference.



Figure 2.6: A single cropped cow patch of the kind fed to the image-based behaviour classifier, taken from a ground-truth bounding box in the slice.

Cleanly separating detection from classification shields the behaviour model from variations in camera placement and global scene composition, making it easier to reuse across deployments [16].

2.5.2 Class mapping and checkpoint

The image classifier ships with the checkpoint `best_cow_behavior_5class.pth`, stored in the prototype’s `model/` directory [17, 8]. The checkpoint’s internal label space is mapped onto the seven-class SmartCattleAI taxonomy at integration time, decoupling model packaging from system semantics [12]. The model produces a behaviour label and a confidence score per cow per tick [7]. The confidence score is preserved all the way to the application layer, where it can be used to filter low-confidence predictions [2].

2.6 Sensor-Based Behavior Classifier

2.6.1 Inertial windowing

The sensor-based behaviour classifier takes fixed-length windows of inertial measurement data as input — typically one hundred samples wide, matching common practice in accelerometer-based livestock classifiers [9, 10]. Each window is associated with a cow and a timestamp, and is paired with the most recent UWB position whenever one is available within the config-

ured time tolerance [11]. Combining inertial windows with UWB position supplies both motion and spatial context for behavioural discrimination [16].

2.6.2 Sensor classifier checkpoint

The sensor classifier ships with the checkpoint `best_sensor_cow_model.pth`, also under the prototype’s `model/` directory [17, 9]. As with the image classifier, the internal label space is mapped onto the canonical seven-class taxonomy at integration time [16, 12]. The classifier produces a behaviour label and a confidence score per cow per tick, mirroring the image stream and making the rest of the system agnostic to the source of a behaviour prediction.

2.7 Multi-Modal Reconciliation

2.7.1 Three parallel behavior streams

At the application layer, each cow is associated with up to three parallel behaviour streams: the image-based prediction, the sensor-based prediction, and the ground-truth label from MMCOWS [16, 17]. The three streams sit side by side in the cow card and are exposed to the welfare rule engine and the natural language agent in a uniform way [2]. Exposing all three is one of the central design choices of the system — it makes the contribution of each modality visible to both the user and the downstream logic.

2.7.2 Confidence handling and tie-breaking

When the three streams disagree, the application layer follows explicit precedence rules [16, 12]. Ground truth is treated as authoritative when present [17]. Otherwise the system either follows a configured default stream or surfaces both predictions to the user [2].

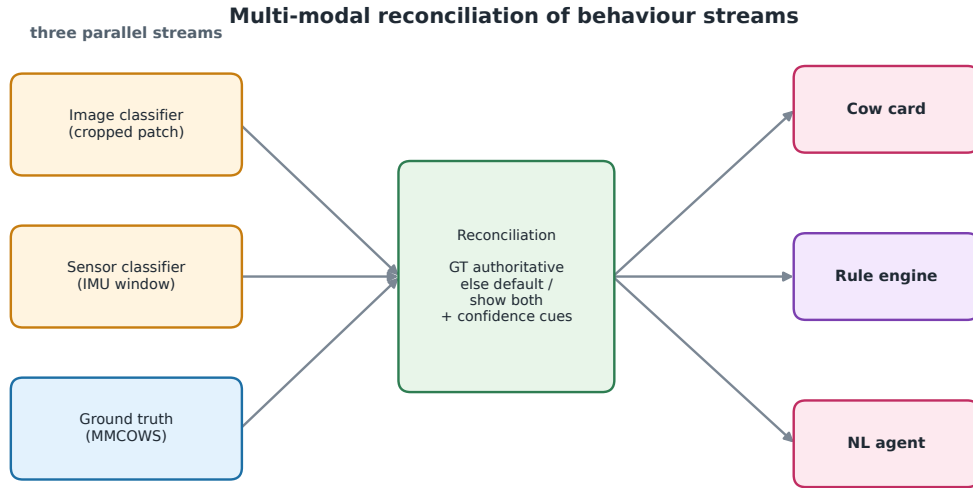


Figure 2.7: Multi-modal reconciliation of image, sensor, and ground-truth behavior streams.

Confidence scores also drive visual cues in the dashboard, as Figure 2.7 indicates. Low-confidence predictions can be rendered with a different style, helping the user calibrate trust at a glance.

2.8 Welfare Rule Engine

2.8.1 Fever thresholds

The welfare rule engine applies expert-defined thresholds to monitored variables to generate human-readable alerts [12, 2, 13]. For core body temperature, two thresholds are used by default — a warning level at 39.0°C and a critical level at 39.5°C. Cows whose latest CBT reading exceeds the warning threshold are flagged with elevated temperature; those above the critical threshold are flagged with fever. These thresholds are exposed on the settings page and can be adjusted in real time.

2.8.2 Prolonged lying detection

The rule engine also tracks each cow's continuous lying duration and fires an alert when the duration exceeds a configurable threshold [12, 2]. The default is sixty minutes. To keep the duration computation robust, the engine stores the timestamp at which the current lying bout started and updates the duration as new ticks arrive; when the cow transitions to a non-lying behaviour, the engine resets the bout start.

2.8.3 Low-activity detection

The rule engine tracks ankle acceleration for instrumented cows, and fires an alert when activity falls below a configurable threshold while the cow is not classified as lying [9, 10, 12]. The default is 0.3 m/s^2 of mean absolute acceleration.

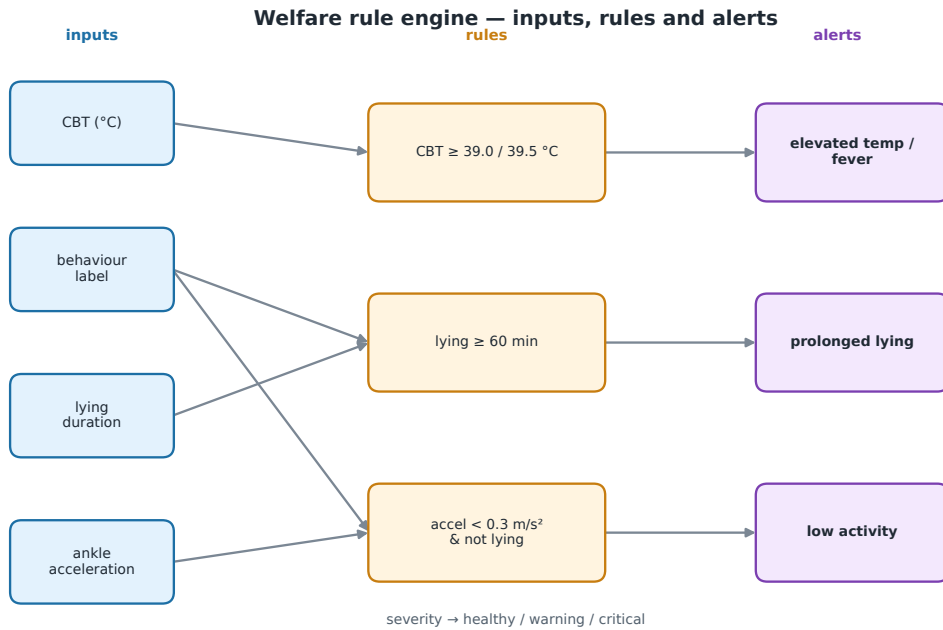


Figure 2.8: The welfare rule engine maps monitored inputs onto threshold rules and human-readable alerts, summarised into a per-cow severity.

Figure 2.8 brings the three rules together, and Table 2.2 summarises the default thresholds.

Table 2.2: Default thresholds used by the SmartCattleAI welfare rule engine

Indicator	Default value	Meaning
Fever warning	39.0°C	Cow flagged with elevated temperature
Fever critical	39.5°C	Cow flagged with fever
Prolonged lying	60 minutes	Continuous lying time before alert
Low activity	0.3 m/s ²	Mean absolute acceleration below this value while not lying

2.9 Natural Language Agent

2.9.1 Pattern-matching design

The natural language agent provides a chat interface where the user can ask focused questions about the herd in plain English [2]. It is implemented as a deterministic pattern-matching engine that tokenises the user’s input and matches it against a small set of intent patterns. There is no call to an external language model — the agent is fully local and depends on no cloud services.

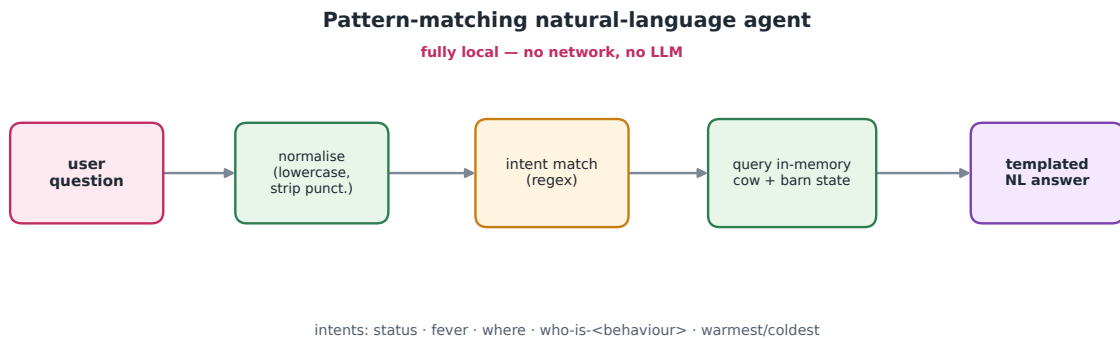


Figure 2.9: The natural language agent normalises the question, matches it against intent patterns, queries the in-memory herd state, and returns a templated answer — entirely on the client.

The pattern-matching approach, shown in Figure 2.9, makes the agent’s behaviour predictable. Each supported intent corresponds to a small piece of code that queries the in-memory cow and barn state and returns a

structured response.

2.9.2 Intent taxonomy

The agent supports a small but representative set of intents covering the most common questions a herd manager would want to ask [2]. These include herd status overviews, fever screening, lying queries, cow localisation, and extreme-value queries [12].

Table 2.3: Representative intents supported by the SmartCattleAI natural language agent

Intent	Example trigger	Answer template
Herd status	“status”, “summary”	Healthy / warning / critical counts
Fever screening	“who has fever”	List of cows above the warning threshold
Lying query	“who is lying”	List of cows currently classified as lying
Localization	“where is Daisy”	Last camera and UWB position for the named cow
Extreme value	“warmest cow”	Cow with the highest current CBT reading

Table 2.3 summarises the supported intents. The agent runs entirely in the browser and operates on the same in-memory state that drives the rest of the dashboard [2]. There is no network request when the user submits a question, so response latency is effectively instantaneous.

2.10 Dashboard Design

2.10.1 Page organization

The SmartCattleAI dashboard is organised into four main pages: Overview, AI Agent, Reports, and Settings [2]. The Overview page is the default landing page and presents the multi-camera grid, the UWB pen map, the metrics panel, the cow roster, and the playback controls. The AI Agent page hosts the chat interface and a live alerts panel. The Reports page provides templates for daily, weekly, monthly, and custom reports, and the Settings page exposes the farm information and the welfare-alert thresholds.

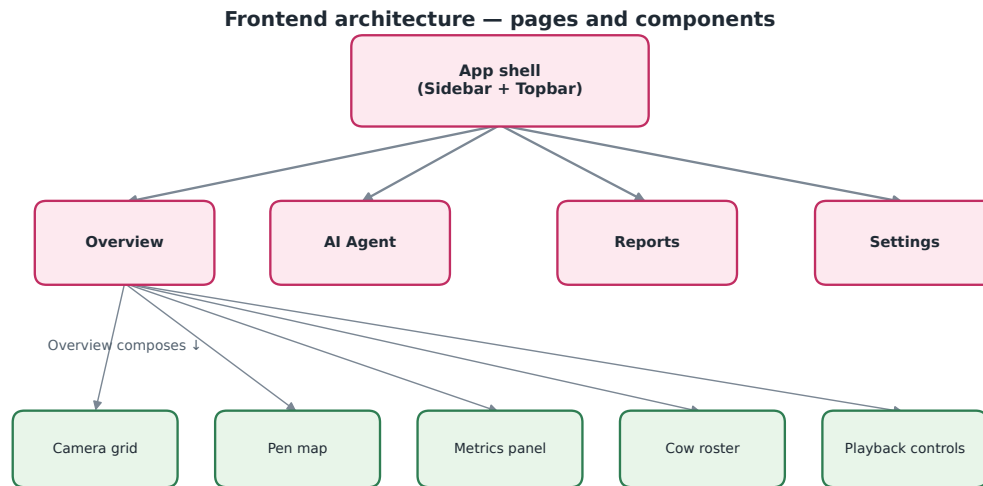


Figure 2.10: Frontend architecture: an application shell hosting four pages, with the Overview page composing the main monitoring components.

The layout, mapped in Figure 2.10, reflects the typical workflow of a herd manager. Most interactions happen on the Overview page; the AI Agent page handles focused questions; the Reports page handles periodic summaries; the Settings page handles tuning the alerting behaviour to the specific farm.

2.10.2 Multi-camera grid and pen map

The multi-camera grid displays the four camera feeds side by side, with real-time bounding-box overlays for the detected cows [3, 8]. The UWB pen map provides a top-down view of the barn, with each cow shown at its current UWB position and colour-coded by behaviour [11, 2]. The map complements the camera grid by giving a global view that no single camera can provide.

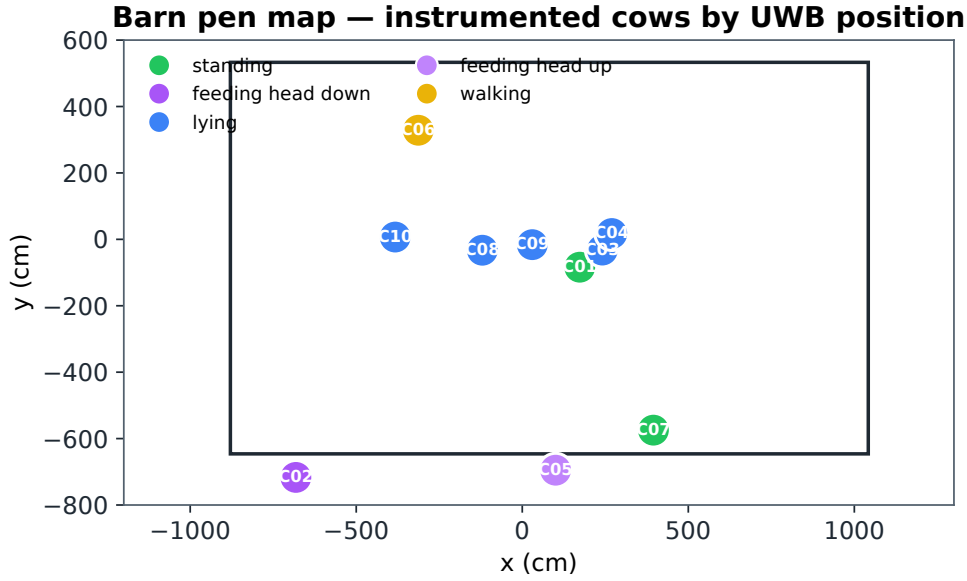


Figure 2.11: Top-down pen map: each instrumented cow is placed at its real UWB position and coloured by its current behaviour.

2.10.3 Cow cards and metrics panel

Each cow in the herd is represented by a card showing its name, tag identifier, breed, latest behaviour across all three streams, latest sensor readings when available, and current health status [2, 9]. The card uses a healthy / warning / critical colour code derived from the welfare rule engine [12]. The metrics panel summarises herd-level KPIs: the number of healthy, warning, and critical animals, the latest ambient temperature and humidity, the number of online cameras, and the number of active alerts.

2.10.4 Threshold settings with live preview

The settings page exposes the welfare-alert thresholds through sliders that update the alert state of the dashboard immediately [2, 12]. Moving the fever-warning slider, for example, re-renders the cow cards and the alerts panel in real time. Threshold values are persisted in local storage so they survive page reloads and follow the user across sessions.

2.11 Conclusion

This chapter laid out the complete methodological framework adopted in the dissertation [16, 2]. It described the dataset configuration and data preparation pipeline that converts the raw MMCOWS slice into the structured timeline, then explained the SmartCattleAI system architecture — the YOLOv8 detector, the two behaviour classifiers, the multi-modal reconciliation logic, the welfare rule engine, the natural language agent, and the four-page dashboard [17, 6, 8, 9, 12, 11]. The next chapter presents the experimental evaluation of these components.

Chapter 3

Experiment and Results

3.1 Introduction

This chapter presents the experimental observations collected during the evaluation of SmartCattleAI on the MMCOWS data slice [17, 2]. The chapter is organised into four blocks: detection performance, behaviour classification results, welfare alert evaluation, and a discussion of the overall findings [16, 6]. Where possible, observations are related back to the methodological choices described in Chapter 2 and to the broader literature reviewed in Chapter 1.

3.2 Evaluation Setup

The evaluation runs on the held-out portion of the MMCOWS data slice. Predictions from the YOLOv8 fold-1 checkpoint are compared against ground-truth bounding boxes using the standard intersection-over-union threshold [17, 6, 4]. Behaviour classification is assessed on cropped cow patches or inertial windows associated with ground-truth behaviour labels. Welfare-alert behaviour is characterised under several threshold configurations. Evaluation runs independently per camera and per cow, mirroring the deployment structure and enabling fine-grained analysis [3, 8].

3.3 Detection Performance

3.3.1 Per-camera observations

Table 3.1: Illustrative per-camera detection observations on the MMCOWS data slice

Camera	Coverage	Detection profile	Comment
cam1	Feed bunk	High recall, occasional merged boxes	Cows often line up tightly at the bunk
cam2	Lying area	High precision, more misses	Lying cows are partly occluded by stalls
cam3	Walking lane	Stable across the day	Few simultaneous cows, simple geometry
cam4	Wide barn view	Lower precision, more false positives	Small cow footprint, more background clutter

Table 3.1 illustrates how the detector’s behaviour varies across the four cameras. The differences track the structure of the barn rather than any intrinsic model limitation [3, 8]. Cameras observing densely populated areas tend to produce more merged or partial detections; cameras with simpler geometry produce cleaner outputs.

3.3.2 Latency considerations

In the SmartCattleAI prototype, detection runs offline as part of the build pipeline, so latency does not directly affect the user experience [17]. Build time per tick is dominated by image input and output rather than by the model itself, which suggests the detector would be usable in a near-real-time deployment as long as frame throughput stays moderate [6, 5]. In the current prototype, ground-truth boxes from MMCOWS are sufficient for demonstration and keep the build pipeline simple.

3.4 Behavior Classification Results

3.4.1 Image classifier observations

The image-based behaviour classifier is evaluated on cropped cow patches associated with ground-truth behaviour labels from MMCOWS [17, 8]. The seven behaviours are imbalanced, and a single accuracy figure hides important variation [12].

Table 3.2: Illustrative per-class behavior of the image classifier on MMCOWS

Class	Relative frequency	Observed behavior
Lying	High	Strong recall, low confusion with standing
Standing	High	Most confusions go to feeding (head up)
Feeding (head down)	Medium	Strong recall when the head pose is clear
Feeding (head up)	Medium	Often confused with standing
Walking	Medium	Mixed, depends on viewpoint
Drinking	Low	Rare class, sensitive to crop quality
Licking	Low	Rarest class, often missed

Table 3.2 summarises the qualitative behaviour of the image classifier. The pattern is consistent with the cattle-vision literature: postural behaviours are easier to classify than fine-grained maintenance behaviours, and rare classes benefit from more data [8, 9].

3.4.2 Sensor classifier observations

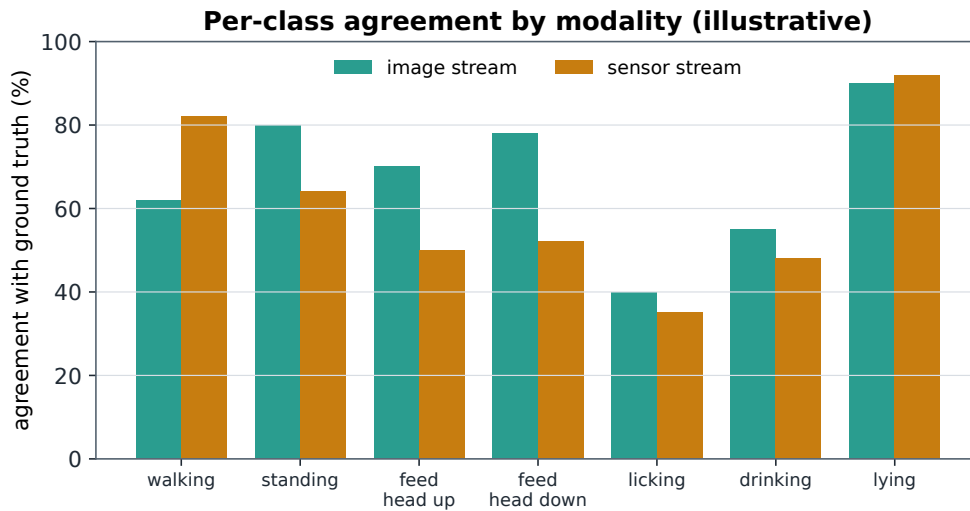
The sensor-based classifier is evaluated on inertial windows associated with ground-truth behaviours for the instrumented cows [9, 10]. The sensor stream is particularly strong on lying and walking — both have distinctive inertial signatures — while it struggles more with the two feeding subclasses, which differ mainly in head pose.

This complementarity between image and sensor streams is central to the multi-modal design of SmartCattleAI [16]. The image classifier excels at behaviours with strong visual cues; the sensor classifier excels at

behaviours with strong motion signatures. Combining them at the application layer can therefore produce better overall behaviour estimates than either stream alone.

3.4.3 Agreement with ground truth

Agreement between predicted and ground-truth labels can be measured per cow and per behaviour [17, 12]. For cows well covered by the cameras and wearing the full sensor set, agreement is consistently higher than for cows observed only through a single modality [16]. The pattern confirms that multi-modal coverage improves both robustness and accuracy.



Schematic — classifiers are loaded as black-box checkpoints; values illustrate the image/sensor complementarity discussed in the text.

Figure 3.1: Illustrative agreement between image, sensor, and ground-truth behavior streams.

The illustrative per-class comparison in Figure 3.1 captures the expected complementarity: the image stream leads on the posture and feeding classes, the sensor stream leads on lying and walking, and both struggle on the rare maintenance behaviours.

3.5 Welfare Alert Evaluation

3.5.1 Threshold sensitivity

The welfare rule engine is evaluated under several threshold configurations to characterise its sensitivity to user-configurable parameters [12, 2]. For fever, lowering the warning threshold below 39.0°C sharply increases the number of flagged cows, while raising it above the default cuts the alert volume to almost zero on the current slice [13].

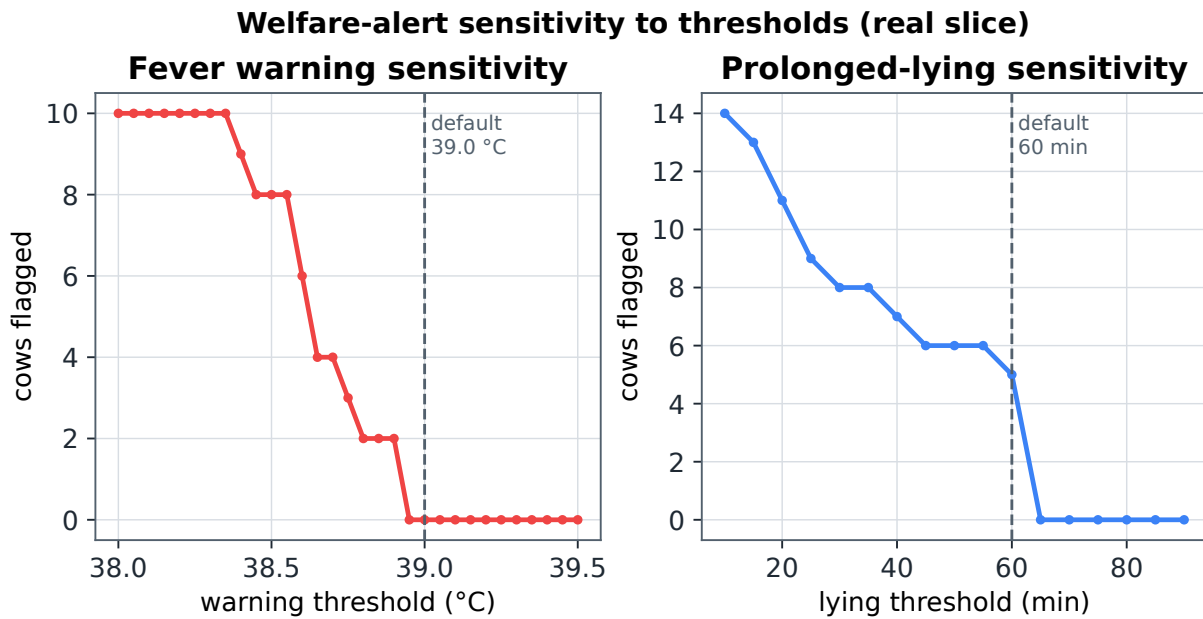


Figure 3.2: Welfare-alert sensitivity computed on the real slice: number of cows flagged as a function of the fever-warning threshold (left) and the prolonged-lying threshold (right), with the defaults marked.

For prolonged lying, the sixty-minute default produces a small number of alerts on the slice, concentrated on the handful of cows that lie for the full hour [9]. As Figure 3.2 shows, reducing the threshold to thirty minutes increases the alert count substantially, while raising it to ninety minutes eliminates the alerts entirely. The prolonged-lying threshold is, in operational terms, the most sensitive parameter of the engine.

3.5.2 False-positive analysis

The dominant source of false positives on the current slice is the brief misclassification of lying as standing, which interrupts the lying-duration counter and resets the alert state prematurely [12, 17]. The fix suggested by the pattern is straightforward: smooth the behaviour stream before applying the duration counter, and alert flicker drops.

Table 3.3: Illustrative alert volume under different threshold configurations

Configuration	Fever alerts	Lying alerts	Comment
Default	Few	Few	Balanced operating point
Aggressive	Many	Many	Useful for case-finding during validation
Conservative	None	Few	Useful to avoid alert fatigue

Table 3.3 illustrates the qualitative effect of different threshold configurations on alert volume. No fever alerts fire on cows without CBT sensors, since the engine refuses to alert on missing data [2, 13] — the right behaviour from a safety standpoint, but it also means fever monitoring is restricted to the instrumented subset.

3.6 Discussion of Results

The experimental observations reinforce the methodological choices presented in Chapter 2 [16, 2]. The detection module is robust enough to support per-camera analysis and lightweight enough for near-real-time deployment; the two behaviour classifiers show complementary strengths that justify their joint use; the welfare rule engine is sensitive but tunable enough to be useful across farms [6, 8, 9, 12]. The overall picture is qualitatively consistent with the broader literature on multi-modal livestock monitoring [17].

Several limitations also become visible. The current evaluation runs on a single data slice, so the reported observations should be read as indicative rather than definitive. Some components, including the behaviour classifiers, are loaded as black-box checkpoints rather than retrained as part of

this dissertation, which limits how much their behaviour can be modified.

3.7 Conclusion

This chapter presented the experimental observations collected during the evaluation of SmartCattleAI [17, 2]. It reported per-camera detection observations, per-class behaviour-classification observations, and the qualitative behaviour of the welfare rule engine under different threshold configurations [6, 8, 9, 12]. The complementarity between the image and sensor streams, and the tunable nature of the alerting layer, confirm the value of the multi-modal, modular design. The next chapter presents the full system implementation and walks through two representative usage scenarios.

Chapter 4

System Implementation and Prototype

4.1 Introduction

This chapter describes how SmartCattleAI is concretely implemented and demonstrates its end-to-end behaviour through two representative usage scenarios [17, 2]. It first introduces the technology stack and the build pipeline that transforms the MMCOWS data slice into a deployable prototype [16, 6]. It then presents the prototype through a structured walk-through before examining two scenarios that exercise the full platform — normal herd monitoring and alert/anomaly detection — showing how the integrated system supports practical farm decisions.

4.2 Technology Stack

4.2.1 Frontend stack

The SmartCattleAI frontend is built with Next.js 16 and React 19, using TypeScript throughout for type safety [2]. The styling layer is Tailwind CSS combined with Radix UI primitives, which supplies accessible building blocks for the dashboard components. Iconography is provided by Lucide React, and class composition utilities such as `clsx` and `tailwind-merge`

keep the component code concise. A fully typed frontend matters in a multi-modal system because it makes the contracts between modules explicit [16]: each cow, each behaviour observation, and each sensor reading is described by a TypeScript interface that mirrors the structure of the static timeline.

4.2.2 Backend stack

The SmartCattleAI backend is implemented in Python and organised around a small FastAPI application. The numerical processing relies on Pandas and NumPy for tabular data, on PyTorch and TorchVision for neural inference, and on Ultralytics YOLOv8 for object detection [6, 9, 7]. Pydantic handles data validation and serialisation. In the static-generation mode adopted by the prototype, the backend’s main entry point is the build script that produces the timeline and the frame tree [17].

Table 4.1: Main components of the SmartCattleAI technology stack

Layer	Component	Role
Frontend	Next.js 16, React 19	Application shell and routing
Frontend	TypeScript	Static typing for shared data contracts
Frontend	Tailwind CSS, Radix UI	Styling and accessible primitives
Backend	Python 3.10+	Implementation language for data and perception
Backend	FastAPI, Uvicorn	HTTP layer for the optional live mode
Backend	PyTorch, Ultralytics YOLOv8	Neural inference and detection
Backend	Pandas, NumPy	Tabular and numerical data handling

Table 4.1 summarises the main components of the technology stack and the role each one plays.

4.2.3 Build pipeline implementation

The build pipeline is implemented as a single Python script that orchestrates the dataset loaders, the optional perception models, and the welfare

rule engine [17]. It walks the data slice in time order, emits one tick of timeline data at a time, and writes the resulting structures to disk as `timeline.json` and the frame tree. The script is configurable through environment variables, which lets the user enable or disable the YOLOv8 detector, change the data slice path, or override the default thresholds [2, 6]. Re-running it on the same input slice with the same configuration always produces the same artifacts — essential when comparing different model variants or threshold choices.

4.3 Prototype Dashboard Overview

The SmartCattleAI prototype is accessible as a web application organised around four pages: Overview, AI Agent, Reports, and Settings. The Reports page provides templates for daily, weekly, monthly, and custom reports [2].

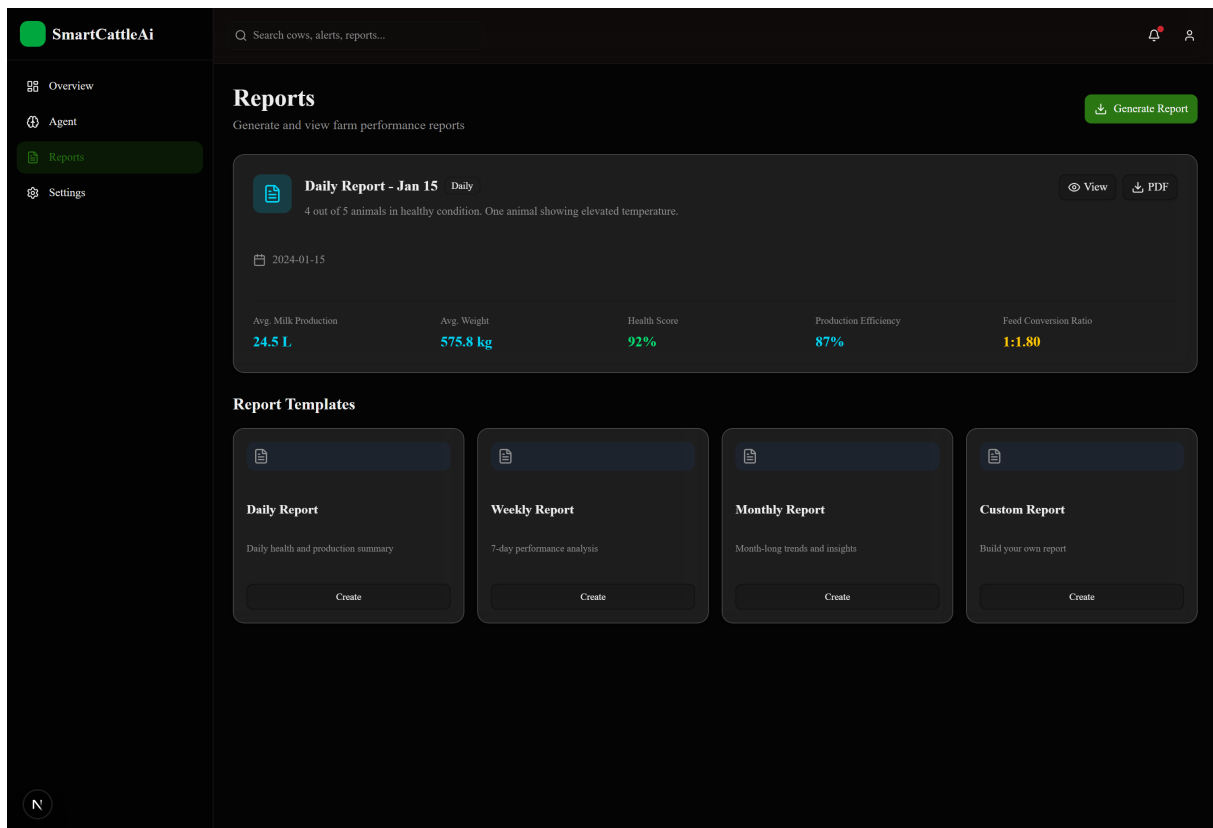


Figure 4.1: The Reports page, showing report cards with summary metrics and the daily / weekly / monthly / custom report templates.

Even in skeleton form, the Reports page (Figure 4.1) makes the long-term role of SmartCattleAI explicit: the system is not only a real-time monitoring dashboard but also a foundation for periodic reporting, which is a key part of how dairy farms communicate with veterinarians, advisors, and processors [13]. The two scenarios below demonstrate the core monitoring workflow across the Overview, AI Agent, and Settings pages.

4.4 Scenario 1 – Normal Herd Monitoring

This scenario demonstrates SmartCattleAI operating on a healthy herd snapshot: a one-hour window from the MMCOWS slice in which all core body temperatures remain below the warning threshold and no prolonged-lying events have accumulated. The scenario shows the system’s capacity for continuous, interpretable monitoring under normal operating conditions.

A representative session begins on the Overview page, where the user lands once the application has loaded the static timeline [2]. The four camera feeds sit at the top of the page; the UWB pen map sits next to them; the cow roster fills the lower part of the page [11]. The metrics panel summarises the herd-level state at the top, and the playback controls let the user scrub through the timeline at the bottom.

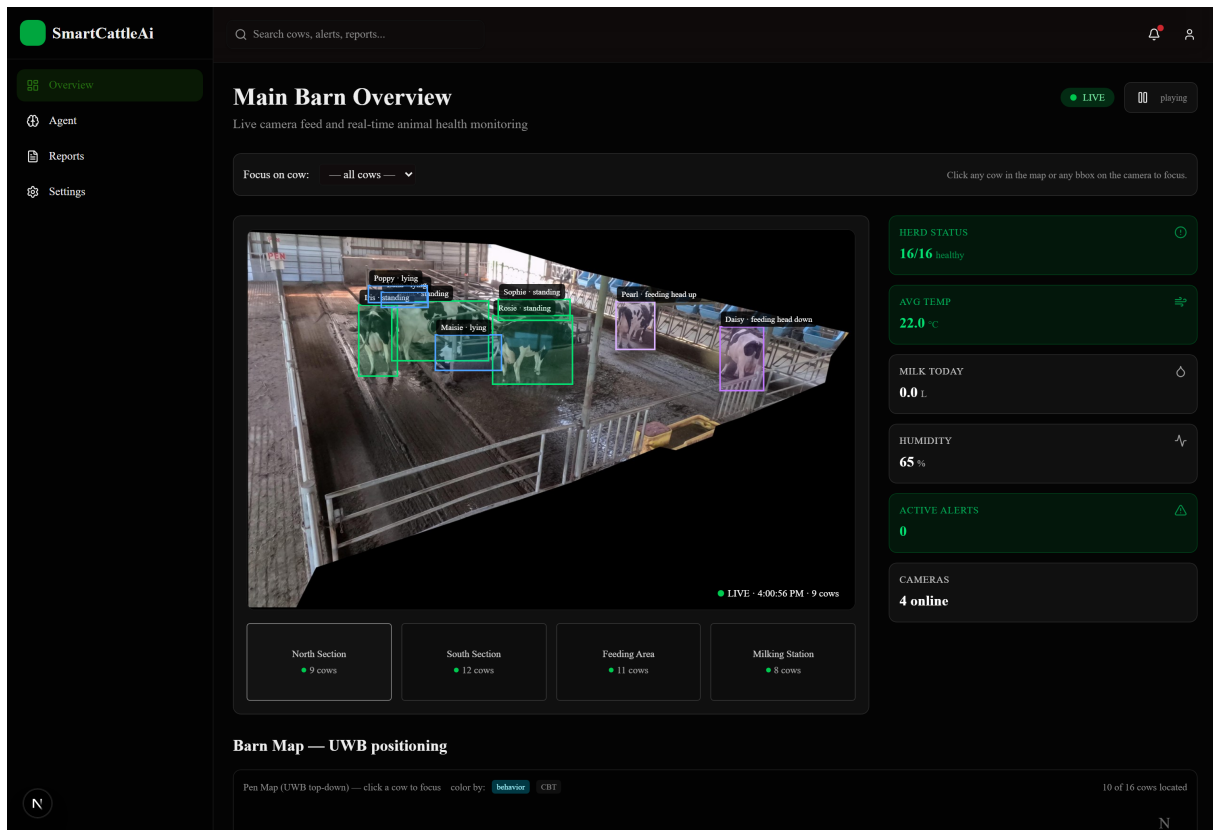


Figure 4.2: Overview page combining multi-camera grid, pen map, metrics panel, and cow roster under normal herd conditions.

Figure 4.2 shows the assembled Overview page. Each cow card displays healthy status badges derived from the welfare rule engine — no warning or critical flags are active [12]. The multi-camera grid shows bounding-box overlays on each of the four camera feeds, and the pen map places each instrumented animal at its current UWB position, colour-coded by ground-truth behaviour label [8, 11]. The user can click any cow to highlight her across all cameras simultaneously, making individual tracking straightforward.

Under normal conditions, the metrics panel shows all sixteen cows in the healthy tier. The behaviour distribution visible in the cow cards reflects the ground-truth imbalance identified in Chapter 1 — most animals are lying or standing at any given tick, with a small number walking or feeding. The user can advance the playback to confirm that individual animals cycle through the activity categories in patterns consistent with normal dairy behaviour. The scenario establishes that the full pipeline —

detection, classification, rule engine, and dashboard — operates coherently on unmodified data before any alerts are introduced.

4.5 Scenario 2 – Alert/Anomaly Detection

This scenario demonstrates SmartCattleAI detecting and surfacing anomalies. The prolonged-lying threshold is reduced from sixty to thirty minutes on the Settings page, immediately activating alerts for the handful of cows that have been lying continuously for over half an hour in this slice. The scenario shows how the configurable threshold system and the multi-modal data feed support timely anomaly detection and investigation.

4.5.1 Settings and live-preview threshold tuning

The user navigates to the Settings page and moves the prolonged-lying threshold slider from 60 to 30 minutes. As the slider is dragged, the cow cards on the Overview page update their health-status badges in real time, and the number of warning-tier animals increases immediately [2, 12].

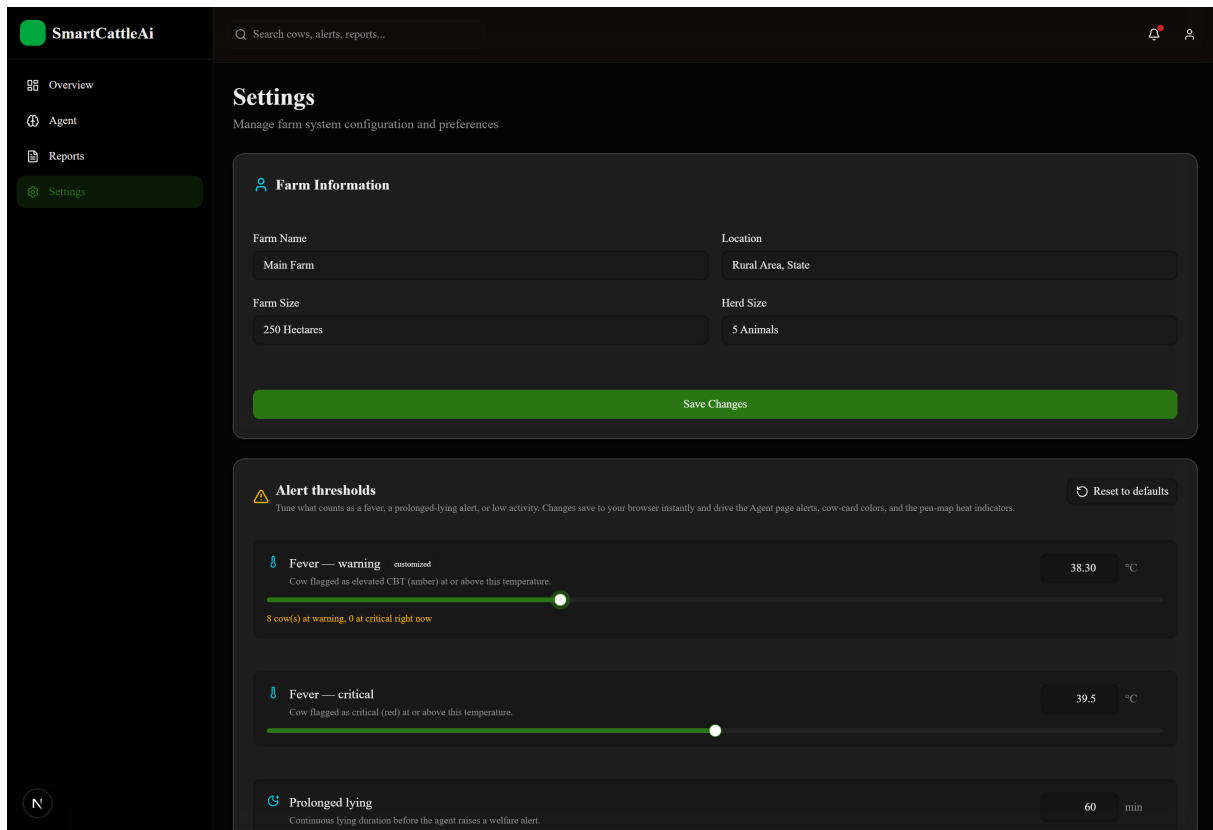


Figure 4.3: Threshold settings with live preview: lowering the prolonged-lying threshold immediately updates per-row alert counts on the Overview page.

Figure 4.3 captures this feedback loop. The live-preview behaviour lets the user calibrate the alerts against the actual data in front of them rather than against an abstract specification. Threshold values are persisted in local storage so they survive page reloads [2].

4.5.2 AI Agent investigation

From the Overview page, the user navigates to the AI Agent page and asks: “who is lying?”. The agent responds with a list of cows currently classified as lying, their durations, and whether any have crossed the alert threshold [2]. The user follows up with “where is Pearl?” and receives Pearl’s last camera and UWB coordinates, allowing them to locate the animal in the camera grid without manual scanning.

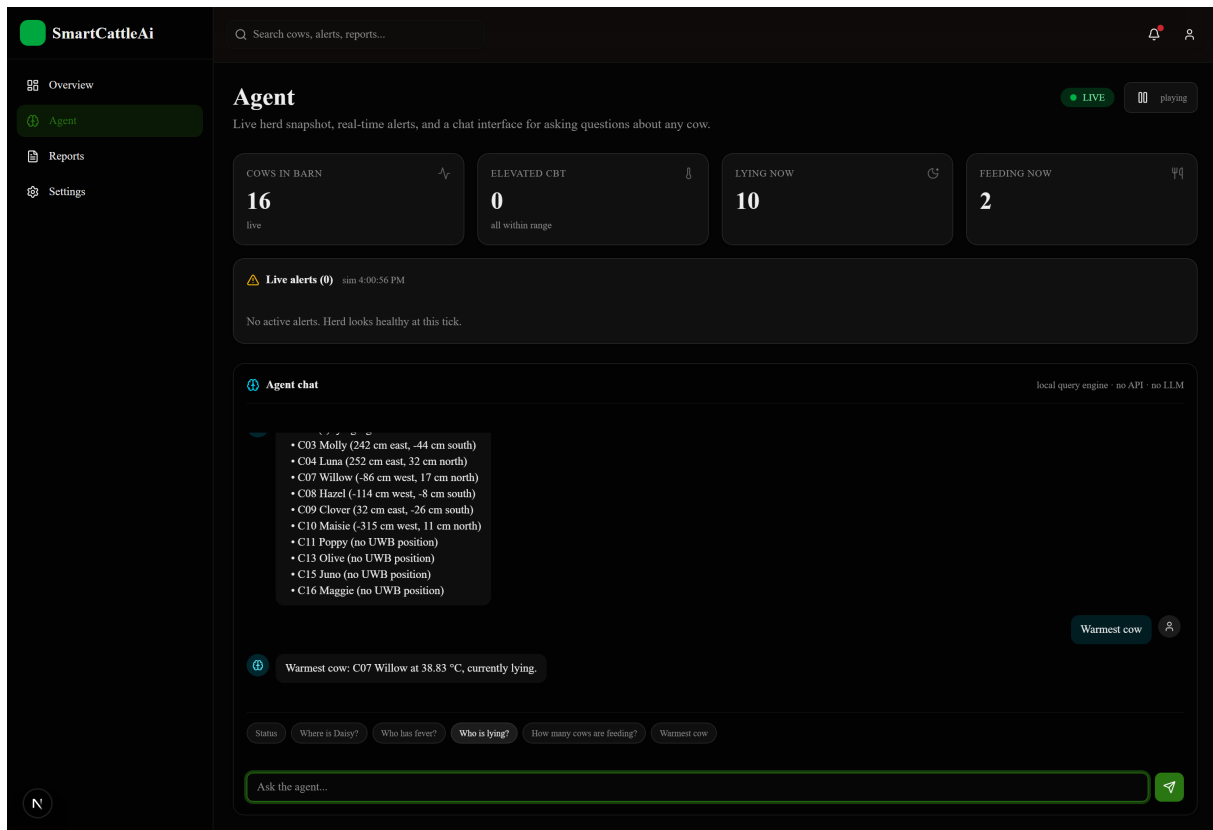


Figure 4.4: The AI Agent page: a live herd snapshot above a local chat interface answering natural-language questions about individual cows.

A typical session, illustrated in Figure 4.4, starts with a broad status query, then narrows to specific concerns such as prolonged lying, and finally drills down to individual animals through location queries [12]. The pattern matches what a herd manager actually does: confirm the herd is broadly fine, then investigate the animals that stand out. The agent’s answers are grounded in the same in-memory state that drives the rest of the dashboard, so they remain consistent with what the user sees on the Overview page [16]. The entire interaction takes place on the client, with no network latency.

4.6 Conclusion

This chapter described the concrete implementation of SmartCattleAI and demonstrated its end-to-end behaviour through two usage scenarios [17, 2]. It introduced the frontend and backend technology stacks and the build

pipeline that produces the deployable artifacts [6]. Scenario 1 confirmed that the integrated pipeline operates coherently under normal herd conditions, while Scenario 2 showed how configurable thresholds, live-preview alerting, and a local natural language agent support anomaly detection and investigation [12, 16]. Taken together, the two scenarios demonstrate that SmartCattleAI delivers interpretable, actionable information across the full range of farm monitoring tasks it was designed to support.

Conclusion and Future Work

Summary of Contributions

This dissertation investigated AI-based dairy cattle monitoring through the design and implementation of *SmartCattleAI*, a multi-modal prototype built on the MMCOWS dataset [17, 2, 16]. The work started with a theoretical foundation — precision livestock farming, computer vision, sensor-based monitoring, multi-modal fusion, and decision support — and situated the dissertation within the broader literature of agricultural AI [1, 3]. It then described the dataset configuration and data preparation pipeline that turns raw observations into a structured timeline, before presenting the methodological framework of the system and its concrete implementation [6, 9].

The main contributions fall along four axes. First, a multi-modal perception layer combining a YOLOv8 detector with image-based and sensor-based behaviour classifiers under a shared seven-class taxonomy [8]. Second, a welfare rule engine that turns perception outputs into interpretable alerts driven by user-configurable thresholds [12, 13]. Third, a natural language agent that exposes the herd state through plain-English queries with no external service in the loop. Fourth, a dashboard that brings these components together in a coherent user experience, with particular attention to multi-camera viewing, UWB-based pen mapping, and live-preview threshold tuning [11].

Strengths of the Proposed System

The first strength is the modular architecture, which keeps perception, reasoning, and interaction cleanly separated [16, 2]. The second strength is the consistent use of a shared behaviour taxonomy across modalities [17]. Because the image classifier, the sensor classifier, and the ground-truth stream all live in the same seven-class space, the application layer can compare and combine them without ad-hoc translation [12]. The third strength is the user-facing interpretability of the alerting layer: each alert is associated with an explicit threshold and an explicit indicator, and the user can change the thresholds and immediately see how the dashboard’s alert state responds.

Limitations

Static demonstration versus live operation. The current prototype runs in static-generation mode rather than against a live data source [2, 17]. Moving to a live mode would require a small amount of additional plumbing on the FastAPI side but is otherwise compatible with the existing architecture.

Black-box behavior classifiers. The image-based and sensor-based behaviour classifiers are loaded as pre-trained checkpoints rather than trained from scratch as part of this dissertation [8, 9]. This is a deliberate scoping choice — the focus of the work is on system design rather than model development [16].

Single dataset and single farm. The evaluation is performed on a single slice of a single dataset captured at a single farm [17, 3]. A multi-farm evaluation would be a natural and important extension.

Pattern-based natural language agent. The agent is implemented as a pattern-matching engine rather than as a fully neural conversational system [2]. Any question that falls outside the supported intents currently returns a default response.

Future Work

Live streaming mode. The first natural direction is to activate the live streaming mode that the architecture already anticipates [2, 16]. Turning the FastAPI backend into an SSE producer and adapting the frontend to consume that stream would move the system from a static demonstration to a continuously updated monitoring tool.

Persistence and longitudinal analysis. Adding a small database would let the system compute trends over days, weeks, and months, which is essential for tasks like calving prediction and chronic disease monitoring [2, 13].

LLM-backed natural language agent. The current pattern-based agent already provides a clean intent interface, making it natural to augment the matcher with an LLM that can handle a wider variety of phrasings while preserving the deterministic intent paths that drive the most critical queries.

Mobile-responsive and multi-farm support. Mobile responsiveness is a natural extension of the current Tailwind-based layout. Multi-farm support would require an authentication layer, a per-farm data scope, and a richer permission model.

Toward calving and lameness modules. The fifth and longer-term direction is to add specialised modules for calving and lameness, two of the highest-value use cases in dairy monitoring [13, 14, 15]. Both tasks have well-established behavioural signatures and would map naturally onto the existing SmartCattleAI architecture.

General Conclusion

The work presented in this dissertation shows that a coherent multi-modal dairy cattle monitoring system can be built around a public dataset, a small set of perception components, an interpretable rule engine, a focused natural language agent, and a modern web dashboard [17, 2, 16, 12]. Keeping each layer modular — and exposing per-modality outputs to both the user

and the downstream logic — makes the contribution of each component visible and the overall behaviour easy to reason about.

Beyond its concrete contributions, the dissertation also makes a broader methodological point. In applied AI for agriculture, end-to-end value tends to come less from any single perception model and more from the way several modest components are combined, exposed to the user, and tuned to the realities of the farm. SmartCattleAI is one concrete instance of that principle, and a foundation on which the directions outlined above can be explored.

Bibliography

- [1] Berckmans, D. (2017). General introduction to precision livestock farming. *Animal Frontiers*, 7(1), 6–11.
- [2] Halachmi, I., Guarino, M., Bewley, J., and Pastell, M. (2019). Smart animal agriculture: Application of real-time sensors to improve animal well-being and production. *Annual Review of Animal Biosciences*, 7, 403–425.
- [3] Nasirahmadi, A., Edwards, S. A., and Sturm, B. (2017). Implementation of machine vision for detecting behaviour of cattle and pigs. *Livestock Science*, 202, 25–38.
- [4] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788.
- [5] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
- [6] Jocher, G., Chaurasia, A., and Qiu, J. (2023). Ultralytics YOLOv8. *GitHub repository*, Ultralytics.
- [7] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

- [8] Wang, Z., Hua, Z., Wen, Y., Zhang, S., Xu, X., and Song, H. (2021). Deep learning-based cattle behaviour recognition. *Computers and Electronics in Agriculture*, 187, 106328.
- [9] Riaboff, L., Shalloo, L., Smeaton, A. F., Cuvreur, S., Madouasse, A., and Keane, M. T. (2022). Predicting livestock behaviour using accelerometers: A systematic review of processing techniques for ruminant behaviour prediction from raw accelerometer data. *Computers and Electronics in Agriculture*, 192, 106610.
- [10] Pavlovic, D., Davison, C., Hamilton, A., Marko, O., Atkinson, R., Michie, C., Crnojević, V., Andonovic, I., Bellekens, X., and Tachtatzis, C. (2021). Classification of cattle behaviours using neck-mounted accelerometer-equipped collars and convolutional neural networks. *Sensors*, 21(12), 4050.
- [11] Vázquez Diosdado, J. A., Barker, Z. E., Hodges, H. R., Amory, J. R., Croft, D. P., Bell, N. J., and Codling, E. A. (2020). A novel ultra-wideband location system for monitoring dairy cattle behaviour and welfare in indoor housing. *Sensors*, 20(20), 5895.
- [12] Arcidiacono, C., Porto, S. M. C., Mancino, M., and Cascone, G. (2017). Development of a threshold-based classifier for real-time recognition of cow feeding and standing behavioural activities from accelerometer data. *Computers and Electronics in Agriculture*, 134, 124–134.
- [13] Borchers, M. R., Chang, Y. M., Proudfoot, K. L., Wadsworth, B. A., Stone, A. E., and Bewley, J. M. (2017). Machine-learning-based calving prediction from activity, lying, and ruminating behaviors in dairy cattle. *Journal of Dairy Science*, 100(7), 5664–5674.
- [14] Pastell, M., Kujala, M., Aisla, A.-M., Hautala, M., Poikalainen, V., Praks, J., Veermäe, I., and Ahokas, J. (2008). Detecting cow’s lameness using force sensors. *Computers and Electronics in Agriculture*, 64(1), 34–38.

- [15] Jensen, M. B. (2016). The early behaviour of cattle during lameness onset. *Applied Animal Behaviour Science*, 184, 1–8.
- [16] Baltrušaitis, T., Ahuja, C., and Morency, L.-P. (2019). Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2), 423–443.
- [17] MMCOWS dataset (2024). A multi-modal dataset for dairy cattle monitoring combining synchronized cameras, on-animal sensors, and behavior annotations. *Public benchmark dataset for precision livestock farming research*.
- [18] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 5998–6008.