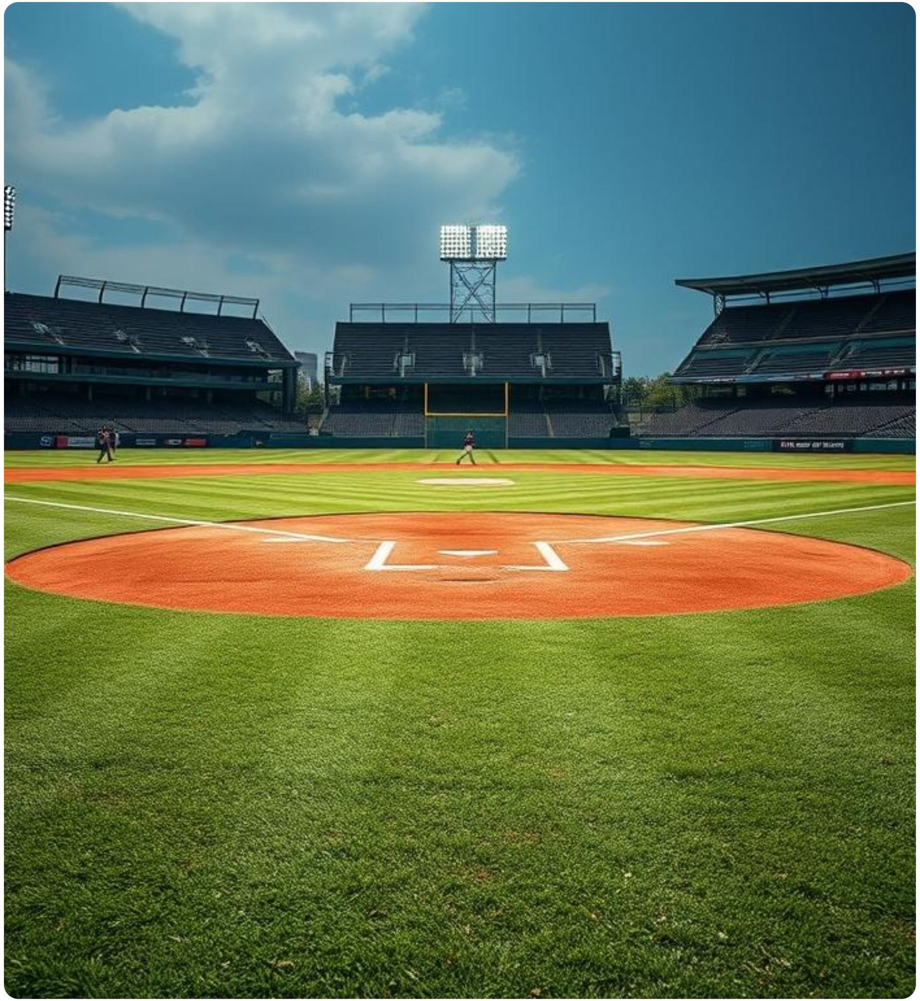
AI-Optimized Container Processing Pipeline for KinaTrax

GCAS: Game-Context Auto-Scaling for Elastic Biomechanical Processing

November 12, 2025



Presenter

About the Presenter

Education & Background

- Bachelor of Science in Game Development, Full Sail University
- Pursuing Master of Science in Computer Science with focus on AI, HCI, and Data Science
- Strong foundation in software engineering and biomechanics

Professional Experience & Motivation

- Senior Software Engineer at KinaTrax (acquired by Sony, October 2024)
- Specializes in biomechanical analysis systems using 8-16 high-speed cameras per event (300+ FPS)
- Experienced in processing massive volumes of real-time sports data
- Passionate about leveraging data to enhance athletic performance
- Motivated to optimize scalable data pipelines for real-time sports analytics

Discovery

The Discovery - A Research Opportunity

Processing Challenges

Current processing takes 6-8 hours post-game due to fixed GPU capacity, causing delays especially during multi-game periods.

Current System Observations

600 events per game captured by 8-16 high-speed cameras generate up to 9,600 videos, constrained by parallel GPU throughput.

Parallelization Opportunity

Cloud elasticity can enable simultaneous processing of all 9,600 videos, reducing processing time from hours to minutes.

Game-Context Intelligence

ML-driven scaling using MLB game states (inning, pitcher changes) predicts processing surges, enabling proactive resource allocation.

Analysis

Current State Analysis

GPU-Constrained Architecture

KinaTrax currently uses fixed-capacity GPU servers located at stadiums, limiting parallel video processing. This infrastructure cannot elastically scale to meet demand spikes.

Processing Workload

Each MLB game generates up to 9,600 videos from 600 events using 8-16 high-speed cameras. Video Generation (VG) takes 4-5 hours, while Track Joint (TJ) processing takes 6-10 minutes per event, running sequentially.

Challenges

Fixed GPU capacity causes resource contention during multiple concurrent games, extending processing times from 6-8 hours to over 12 hours. Hardware upgrades disrupt operations and scaling is cost-inefficient.

Business Impact

Delayed post-game analysis limits coach and scout access to timely insights. Resource inefficiency leads to wasted capacity during low-demand periods, impacting overall operational effectiveness.

Hypothesis

Research Hypothesis

GCAS Innovation Hypothesis

Developing an ML-driven auto-scaling system using MLB game state data can proactively provision cloud GPU resources, eliminating reactive lag and reducing processing latency from hours to minutes.

Key Research Questions

1. Which MLB game state features best predict processing load? 2. How much cost-performance improvement does proactive GCAS provide vs reactive scaling? 3. Can GCAS maintain 100% accuracy correlation in cloud deployment?

Parallelization Efficiency Validation

Validate the speedup potential by testing elastic GPU orchestration with varying container counts (50-500), measuring speedup, efficiency, and cost per event.

Statistical Accuracy Validation

Conduct hypothesis testing to confirm zero accuracy degradation in cloud (Pearson r = 1.0), using Bland-Altman plots and sample of 10-30 games with α=0.05 significance level.

Cost-Performance Optimization

Perform multi-objective optimization to minimize latency and cost while maximizing throughput, focusing primarily on AWS deployment aligned with KinaTrax roadmap.

AI Components - GCAS (Game-Context Auto-Scaling)

GCAS Uniqueness

Unlike traditional reactive cloud scaling based on CPU/memory metrics with 2-5 minute lag, GCAS uses MLB game state inputs to proactively scale resources 30-60 seconds before demand surges.

Game State Features

Key predictive inputs include inning number, score differential, pitcher change flag, base runners count, and high-leverage situation indicator, capturing real-time game context for load forecasting.

ML Model Approach

Phase 1 uses Linear Regression for simplicity and transparency, targeting $R^2 > 0.5$; Phase 2 (stretch goal) explores Random Forest for improved accuracy ($R^2 > 0.7$) and non-linear relationships.

Workflow Overview

Game state changes trigger GCAS ML prediction, which proactively commands Kubernetes to scale containers before workload spikes, ensuring resources are ready upon demand arrival.

Key Benefits

Eliminates 2-5 minute reactive scaling lag, reduces cost by 30-40%, ensures sub-5-minute processing latency for 9,600 videos, and incorporates sports-domain intelligence unavailable in generic cloud platforms.

Example Scenario

When a relief pitcher enters in the 7th inning, GCAS predicts the surge immediately and scales resources within 30 seconds, unlike traditional systems that react only after CPU spikes, causing delays up to 5 minutes.

HCI

HCI Components

Real-Time Processing Dashboard

Displays live status of queued, in-progress, and completed events with clear VG and TJ progress bars and estimated completion times for each event and game.

Priority Request Interface

Offers Rush, Standard, and Economy tiers allocating GPU containers based on urgency and cost, ensuring 100% accuracy while optimizing resource use and cost trade-offs.

Notification System

Supports email, SMS, Slack, and push notifications to alert users of processing completions, delays, or errors, improving communication and responsiveness.

User Research Design

Includes A/B testing of dashboard interfaces, usability testing with 10-12 users, and satisfaction surveys to refine interface effectiveness and user adoption.

Current vs Proposed Performance

Current system processes 9,600 videos in 6-8 hours with no priority options or real-time visibility; proposed system reduces processing to minutes with tiered priorities and live monitoring.

Cost vs Resource Allocation

Higher priority tiers allocate more containers at higher cost (Rush ~2x cost), while Economy minimizes cost with fewer containers; all tiers maintain full accuracy.

Data Science

Data Science Components

GCAS ML Model Development

Develop GCAS using supervised ML models: start with Linear Regression predicting processing load from MLB game state features, then explore Random Forest for improved accuracy and feature importance analysis.

Parallelization Efficiency Analysis

Test processing speedup with varying container counts (50, 100, 200, 500), measuring throughput, efficiency, and cost per event to identify optimal elastic scaling strategies.

Cost-Performance Optimization

Apply multi-objective optimization to balance latency reduction and cost savings, using Pareto frontier analysis primarily on AWS deployments aligned with KinaTrax roadmap.

Statistical Accuracy Validation

Conduct hypothesis testing to confirm zero accuracy degradation in cloud: require 100% correlation (Pearson r=1.0) with on-prem baseline, using Bland-Altman plots and significance tests across 10-30 games.

Cloud Deployment Evaluation

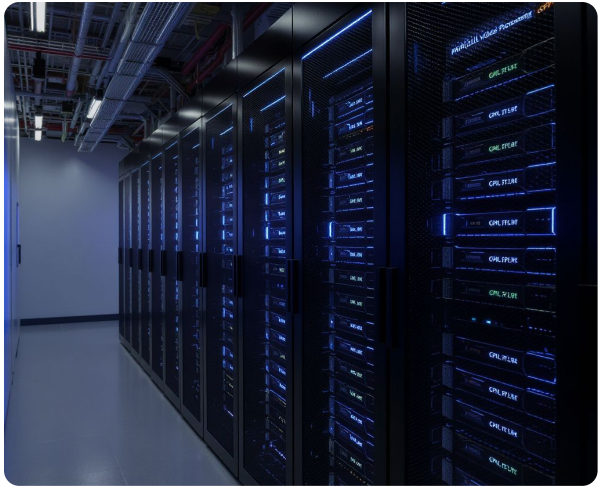
Benchmark existing KinaTrax vision models on cloud GPUs (AWS P3/P4), comparing latency, accuracy correlation, cost, and GPU utilization versus on-prem solutions.

Research Approach & Methodology

Integrate ML modeling, cloud benchmarking, statistical analysis, and cost evaluation with rigorous experimental design to validate GCAS effectiveness and inform scalable cloud migration strategy.

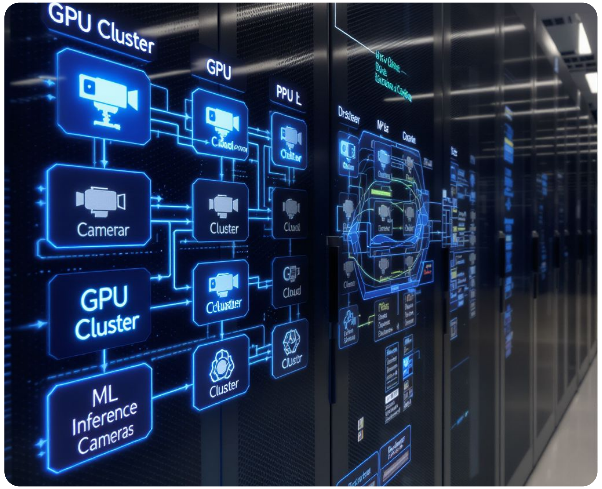
Architecture Overview

Architecture Overview - Current vs Proposed



Current GPU-Constrained Architecture

On-premises fixed GPU servers process up to 9,600 videos per game with limited parallel throughput. Video Generation takes 4-5 hours; Track Joint processing runs in parallel but constrained. Typical completion is 6-8 hours, extending to 12+ hours under multi-game GPU contention.



Proposed Cloud-Scale Elastic Architecture with GCAS

Cloud storage and real-time event streaming feed a Kubernetes cluster with elastic GPU container orchestration. GCAS uses MLB game state ML predictions to proactively scale 0 to 200+ containers, eliminating reactive lag and processing videos in under an hour with maintained accuracy.

Features

Major Features

Elastic GPU Orchestration

Kubernetes-based container management enabling dynamic scaling from zero to over 200 GPU containers, ensuring flexible resource allocation aligned with real-time demand.

GCAS: Game-Context Auto-Scaling

Proactive ML-driven scaling using MLB game state features to predict processing load, eliminating traditional reactive lag and optimizing cost and performance.

Priority-Based Processing

Three-tier priority system (Rush, Standard, Economy) balancing resource allocation and cost without sacrificing accuracy, tailored for high-stakes games or routine analysis.

Real-Time Monitoring Dashboard

Live processing status with VG/TJ progress breakdown, queue visualization, cost tracking, and alert notifications to empower operational transparency and rapid response.

Existing KinaTrax Model Deployment

Cloud deployment of proven markerless motion capture models (OpenPose, HRNet, MediaPipe, YOLO-Pose) maintaining 100% accuracy with GPU-accelerated inference.

Multi-Cloud Ready Architecture

Cloud-agnostic design ensures vendor independence and future flexibility, with AWS as primary focus and optional Azure/GCP support for strategic adaptability.

Architecture

Technical Architecture

Container Orchestration

Kubernetes on AWS EKS manages Docker containers for biomechanical processing. Horizontal Pod Autoscaler integrates with GCAS for dynamic GPU container scaling from zero to 200+ based on predicted load.

GCAS Integration

GCAS ML model runs as Flask API receiving game state inputs to predict load. Kubernetes HPA polls API every 30 seconds to trigger proactive scaling, eliminating reactive lag.

Event Processing & Messaging

Real-time event ingestion via Kafka or AWS Kinesis (TBD). Redis manages priority queues, balancing critical and routine processing for timely resource allocation.

Data Storage & Cloud Deployment

Time-series data stored in InfluxDB, metadata in PostgreSQL, videos in S3-compatible storage. KinaTrax vision models run on AWS P3/P4 GPU instances with TorchServe and TensorFlow Serving.

Monitoring & Observability

Prometheus collects metrics on container health, GPU use, and latency. Grafana visualizes real-time dashboards with VG/TJ progress and cost tracking. Jaeger enables tracing; logs via CloudWatch or ELK stack.

Infrastructure & Security

Infrastructure as Code managed with Terraform or CloudFormation. Security via end-to-end encryption, RBAC, network policies, and secrets management with AWS Secrets Manager or Vault. Architecture remains flexible for Kafka or Terraform choices.

Performance

Success Metrics

Latency Reduction

Reduce processing time from 6-8 hours (typical) and 12+ hours (multi-game) to under 5 minutes for all 9,600 videos, achieving a 99.5% improvement validated by paired t-tests.

Accuracy Preservation

Maintain 100% accuracy correlation (Pearson r = 1.0) with zero degradation, verified by rigorous statistical tests including Bland-Altman plots on 10-30 game samples.

GCAS Effectiveness

Achieve ML model prediction accuracy with MAE < 30 events and $R^2 > 0.5$ (Linear Regression) or > 0.7 (Random Forest), eliminating the 2-5 minute reactive lag to under 30 seconds.

Improved Scalability

Enable processing of 2-3 concurrent games without increased delays, overcoming fixed GPU capacity and resource contention through elastic scaling.

Cost Analysis

Explore potential 30-40% cost reductions via intelligent scaling versus always-on GPU infrastructure, focusing on performance and scalability gains over long-term cost.

User Satisfaction & Bottleneck Identification

Target +40% improvement in user satisfaction (SUS scores) through real-time visibility and priority processing; identify VG vs TJ bottlenecks to guide future optimizations.

Validation

Validation Plan

Proof of Concept

Deploy AWS EKS and containerize pipeline. Develop GCAS v1. Test with 10 historical games. Establish 100% accuracy baseline.

- Cloud system deployed
- Baseline latency & accuracy
- GCAS dataset ready
- Containerized pipeline

Pilot Deployment

Deploy in 1 stadium for 15-20 live games. Collect GCAS data with manual entry. Conduct user research and validate multi-game contention.

- Pilot live deployment
- GCAS performance data
- User feedback reports
- Multi-game validation

Analysis & Reporting

Analyze latency, accuracy, and GCAS effectiveness. Identify VG vs TJ bottlenecks. Synthesize user research. Prepare final report and thesis.

- Statistical analysis
- Bottleneck report
- User research summary
- Final thesis & report

Risk Mitigation & Gates

Monthly phase gates with GO/NO-GO. Maintain 20% schedule buffer. Use fallback data if pilot delayed. Simplify models if needed.

- Phase gate decisions
- Risk dashboard
- Backup plans
- Adaptive scope management

Implementation

Implementation Timeline - 4 Months

Month 1

Foundation & Data Prep

Set up AWS EKS, configure Terraform, containerize pipeline, establish accuracy baseline, collect GCAS data, and run parallelization tests. Thesis work: literature review and background.

Month 2

GCAS Dev & Monitoring

Develop GCAS v1 Linear Regression, integrate Flask API with K8s HPA, set up Grafana dashboard, implement Redis queue, and manual game state entry. Thesis: methods & design docs.

Month 3

Pilot & User Research

Deploy to one stadium, process 15-20 live games, collect GCAS data, conduct user interviews and usability tests, and A/B test dashboard. Thesis: preliminary results & synthesis.

Month 4

Analysis & Thesis Final

Perform statistical analysis, complete performance report, results, discussion, conclusions, abstract, and prepare for thesis defense.

Implementation

Phased Implementation - MVP First

01 **Phase 1: MVP Deliverables**

AWS Kubernetes deployment with GCAS auto-scaling, Docker containerization, GCAS v1 Linear Regression model, Grafana dashboard, priority queuing, single-stadium pilot (15-20 games), user research (10-12 interviews), 100% accuracy validation, and complete thesis.

02 **Phase 2: Enhancement Stretch Goals**

If ahead, develop GCAS v2 Random Forest, automate game state input with Statscast API, build custom React dashboard, expand pilot to multiple stadiums, increase user study size, conduct multi-cloud comparison, and advanced VG/TJ analysis.

03 **Phase 3: Production Rollout Plans**

Post-capstone full GCAS integration with real-time Statscast API, multi-cloud deployment (AWS, Azure, GCP), advanced features like chaos engineering, GraphQL API, mobile dashboards, multi-camera sync, and hybrid cloud with edge support over 6-12 months.

04 **Strategic & Academic Benefits**

Ensures core innovation delivery on time, demonstrates 144x parallelization, maintains 100% accuracy, enables evidence-based cloud migration, offers novel sports-domain ML contribution, and mitigates risks via phased scope & buffers.

Strategy

Competitive Landscape & Strategic Alignment

01

Competitive Alternatives Overview

AWS Wavelength and Azure Edge offer reactive scaling tied to vendor lock-in and high costs. NVIDIA Fleet Command focuses on hardware-specific AI inference without game-context intelligence. Red Hat OpenShift is complex and lacks sports-domain predictive scaling.

02

GCAS Unique Differentiators

Proactive ML-driven scaling based on MLB game state features eliminates 2-5 minute reactive lag found in generic cloud platforms. GCAS integrates sports-domain intelligence, providing a defensible moat and potential IP advantage.

03

Cloud-Agnostic Architecture

Designed with Kubernetes for vendor independence and future flexibility. Supports AWS as primary focus with optional Azure/GCP, ensuring cost optimization and avoiding vendor lock-in while enabling multi-cloud readiness.

04

Proven Biomechanical Algorithms

Deploys KinaTrax's validated markerless motion capture models (OpenPose, HRNet, MediaPipe, YOLO-Pose) in a cloud environment. Ensures 100% accuracy correlation, leveraging years of MLB stadium deployments for reliability.

05

Strategic Alignment with KinaTrax Roadmap

Aligns with KinaTrax's cloud migration plans and Derek's endorsement. Supports scalability for 2027 HEI integration and evolving business needs. Provides evidence-based insights for leadership decisions on multi-cloud and cost-performance trade-offs.

Risk Management

Risk Mitigation

Proactive risk management across technical, business, schedule, alignment, and quality domains ensures project success with clear mitigation strategies and realistic expectations.

Risk Category	Specific Risk	Likelihood	Impact	Mitigation Strategy
Technical	Accuracy degradation in cloud	Medium	High	Extensive testing, 100% correlation requirement, pilot validation
Technical	GCAS prediction accuracy insufficient	Medium	Medium	Start with Linear Regression, fallback to rule-based scaling
Technical	VG bottleneck persists in cloud	Medium	High	Research will identify if VG or TJ is primary constraint, targeted optimization
Business	Multi-cloud scope creep	Medium	Medium	AWS primary focus, Azure/GCP optional pending approval
Business	Cost overruns	Low	High	Budget monitoring, cost alerts, use free tier initially
Business	Stadium availability for pilot	Medium	High	Use historical data if live pilot blocked, flexible scheduling
Schedule	Month 2 overload (GCAS + dashboard)	Medium	High	Simplify GCAS to Linear Regression, use Grafana, 20% schedule buffer
Schedule	Pilot deployment delays	Medium	Medium	Start early, have historical data fallback
Alignment	Technology choices don't match company direction	Low	Low	Maintain flexibility, align with leadership decisions
Quality	100% accuracy not achieved	Low	Critical	Extensive baseline validation, continuous monitoring, GO/NO-GO gates

Conclusion & Action

Key Takeaways & Next Steps

Problem Overview

GPU-constrained parallel processing creates bottlenecks, causing 6-8 hour delays typically and 12+ hours during multi-game periods. Fixed capacity limits scalability and timely insights.

Core Innovation - GCAS

GCAS uses MLB game state and ML to proactively scale cloud resources, eliminating 2-5 minute reactive lags and enabling sub-5-minute processing for all 9,600 videos.

Research Methodology

Integrated AI, HCI, and data science with cloud deployment validation and user research to ensure accuracy, scalability, and user satisfaction for the GCAS system.

Expected Contributions

Achieve speedup via elastic parallelization, maintain 100% accuracy correlation, identify VG vs TJ bottleneck, and provide evidence-based data to guide KinaTrax's cloud migration.

Strategic Alignment

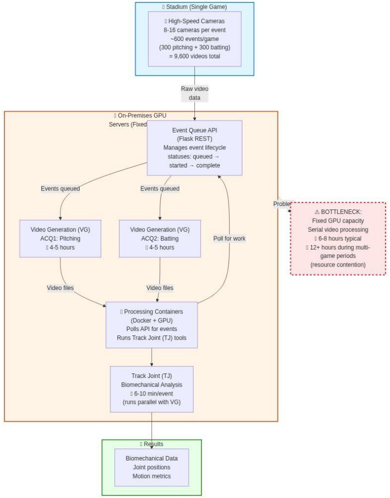
Aligns with KinaTrax's cloud migration roadmap, focuses on AWS primary deployment, supports HEI 2027 integration flexibility, and drives evidence-based strategic decisions.

Next Steps

Confirm leadership approval of MVP scope, finalize AWS budget and pilot stadium access, initiate AWS setup and data collection, and maintain weekly checkpoints with phase gate reviews.

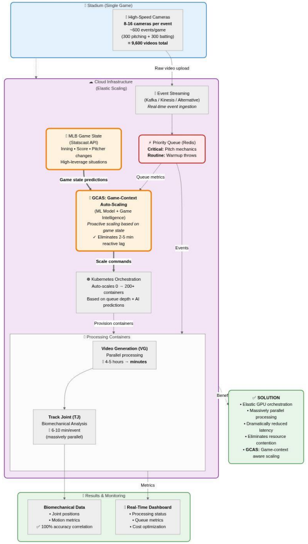
Architecture

Architecture Diagrams (Side-by-Side Comparison)



Current Architecture: Fixed GPU Capacity

On-premises GPU servers process 9,600 videos per game with limited parallelism constrained by fixed GPU capacity. Video Generation (VG) takes 4-5 hours, Track Joint (TJ) 6-10 min/event, resulting in 6-8 hours typical latency, extending to 12+ hours during multi-game periods due to resource contention.



Proposed Architecture: GCAS Cloud-Scale Elasticity

Cloud-native pipeline leveraging Kubernetes for elastic GPU container orchestration driven by GCAS, a game-context aware ML auto-scaler. Proactive scaling adjusts resources 30-60 seconds before demand surges, enabling simultaneous processing of all 9,600 videos in under 5 minutes, eliminating contention and maintaining 100% accuracy.