06 User Stories

2025-10-19

Contents

1 User Stories - Vision-Based Pick and Place System

1.1 Overview

This document contains **user stories** organized by persona, mapped to core robotics concepts, and prioritized using the MoSCoW method (Must have, Should have, Could have, Won't have).

1.2 1. Personas

1.2.1 1.1 Primary Personas

Persona	Role	Goals	Pain Points
Alex (Operator)	Production floor operator	Run system efficiently, monitor status, handle errors	Complex interfaces, unclear error messages
Jordan (Integrator)	Robotics system integrator	Deploy and configure system, calibrate sensors	Poor documentation, difficult setup
Sam (Engineer)	Software/robotics engineer	Develop new features, debug issues, optimize	Lack of modularity, hard-to-trace bugs
Morgan (Manager)	Production/operations manager	Maximize uptime, throughput, ROI	Lack of visibility, slow cycle times
Casey (Maintenance)	Maintenance technician	Perform routine maintenance, diagnose faults	No diagnostic tools, unclear procedures
Taylor (Data Scientist)	AI/ML engineer	Train models, improve detection accuracy	Limited labeled data, model drift

1.2.2 1.2 Secondary Personas

Persona	Role	Goals
Riley (Safety Officer)	Safety compliance manager	Ensure system safety, compliance with standards
Drew (Customer)	End customer/client	Reliable system, good support, fast ROI

1.3 2. User Stories by Persona

1.3.1 2.1 Alex (Operator)

1.3.1.1 Story 1: Basic Operation (Must Have)

As an operator, **I want** to start and stop the pick-and-place system with a single button press, **So that** I can operate the system without technical knowledge.

Acceptance Criteria: - Single "Start" button initiates the full workflow (scan → pick → place) - "Stop" button safely halts all motion within 1 second - Emergency stop button cuts power to motors immediately (<100ms) - Visual indicator (green/red LED) shows system status

Priority: Must Have **Story Points:** 3 **Related Concepts:** State Machine, Task Orchestration, Safety

---

1.3.1.2 Story 2: Real-Time Monitoring (Must Have)

As an operator, **I want** to see a live dashboard showing robot status, throughput, and errors, **So that** I can quickly identify and address issues.

Acceptance Criteria: - Dashboard displays: current state, objects processed, cycle time, error log - Updates in real-time (<1 second latency) - Color-coded alerts (green=OK, yellow=warning, red=error) - Accessible via web browser or touchscreen HMI

Priority: Must Have **Story Points:** 5 **Related Concepts:** UI/Visualization, Monitoring & Logging

---

1.3.1.3 Story 3: Error Recovery Guidance (Should Have)

As an operator, **I want** clear instructions when an error occurs (e.g., "Object not detected - check lighting"), **So that** I can resolve issues without calling engineering support.

Acceptance Criteria: - Error messages are human-readable (no error codes alone) - Suggested recovery actions displayed (e.g., "Retry", "Adjust camera", "Call support") - One-click retry mechanism for transient errors - Error log with timestamp and severity

Priority: Should Have **Story Points:** 5 **Related Concepts:** Error Handling, State Machine

---

1.3.1.4 Story 4: Manual Intervention Mode (Should Have)

As an operator, **I want** to manually jog the robot to a safe position if automatic recovery fails, **So that** I can clear obstructions or reposition objects.

Acceptance Criteria: - Teach pendant or joystick interface available - Robot moves at reduced speed (10% max) in manual mode - Collision detection active during manual jog - Auto-returns to home position after manual intervention

Priority: Should Have **Story Points:** 8 **Related Concepts:** Motion Control, Safety

---

1.3.2 2.2 Jordan (Integrator)

1.3.2.1 Story 5: Guided Calibration (Must Have)

As an integrator, **I want** a step-by-step wizard to calibrate the camera-to-robot transformation, **So that** I can deploy the system without deep robotics knowledge.

Acceptance Criteria: - Wizard guides through: robot positioning, checkerboard placement, image capture - Automatic computation of hand-eye calibration matrix - Validation step (detect known object, verify position accuracy <5mm) - Calibration results saved and version-controlled

Priority: Must Have **Story Points:** 13 **Related Concepts:** Camera-Robot Calibration, Coordinate Transforms

---

1.3.2.2 Story 6: Workspace Configuration (Must Have)

As an integrator, **I want** to define pick and place zones via a GUI (draw polygons in RViz), **So that** I can customize the system for different factory layouts.

Acceptance Criteria: - Interactive zone definition in RViz2 (draw 2D polygons, set heights) - Save/load zone configurations (YAML files) - System only picks/places within defined zones - Collision objects automatically added to planning scene

Priority: Must Have **Story Points:** 8 **Related Concepts:** Motion Planning, Collision Avoidance

---

1.3.2.3 Story 7: Gripper Selection (Should Have)

As an integrator, **I want** to select from pre-configured gripper types (parallel jaw, suction, custom), **So that** I can adapt to different object types.

Acceptance Criteria: - Dropdown menu to select gripper type - System adjusts grasp planning algorithm based on selection - Gripper parameters (max force, stroke) configurable - Test grasp function to validate configuration

Priority: Should Have **Story Points:** 8 **Related Concepts:** Grasp Planning, End-Effector Control

---

1.3.2.4 Story 8: Simulation Before Deployment (Should Have)

As an integrator, **I want** to test the system in Gazebo simulation before running on real hardware, **So that** I can validate configuration without risk.

Acceptance Criteria: - Launch files for both simulation and real hardware - Simulated camera provides synthetic images (textured objects) - Motion planning and control identical in sim and real - "Sim-to-real" checklist ensures parity

Priority: Should Have **Story Points:** 13 **Related Concepts:** Simulation, Testing

---

1.3.3 2.3 Sam (Engineer)

1.3.3.1 Story 9: Modular Architecture (Must Have) **As** an engineer, **I want** the system to use ROS2 nodes with well-defined interfaces (topics, services, actions), **So that** I can develop and test modules independently.

Acceptance Criteria: - Each subsystem (vision, planning, control) is a separate ROS2 package - Interfaces documented in README with message types - Launch files support launching individual nodes for testing - Unit tests for each node (>80% code coverage)

Priority: Must Have **Story Points:** 21 **Related Concepts:** ROS2 Architecture, Software Design

---

1.3.3.2 Story 10: Live Parameter Tuning (Should Have) **As** an engineer, **I want** to adjust PID gains and motion parameters via dynamic reconfigure, **So that** I can optimize performance without recompiling code.

Acceptance Criteria: - ROS2 parameters for: PID gains, trajectory speed limits, detection thresholds - Changes take effect immediately (no restart required) - Parameter server maintains history (rollback support) - Save tuned parameters to YAML for persistence

Priority: Should Have **Story Points:** 5 **Related Concepts:** Control Tuning, ros2_control

---

1.3.3.3 Story 11: Debugging Tools (Should Have) **As** an engineer, **I want** to record ROS2 bags of failed pick attempts and replay them, **So that** I can debug issues offline.

Acceptance Criteria: - Auto-record bags on error (last 30 seconds of data) - Bags include: images, joint states, TF, planning scene - Replay mode allows stepping through time - Annotate bags with error descriptions

Priority: Should Have **Story Points:** 8 **Related Concepts:** Logging, Debugging

---

1.3.3.4 Story 12: Custom AI Model Integration (Could Have) **As** an engineer, **I want** to swap the object detection model (e.g., YOLOv8 → custom model), **So that** I can adapt to new object types.

Acceptance Criteria: - Model loader accepts ONNX or TensorRT formats - Configuration file specifies model path, input size, classes - Inference node automatically adjusts preprocessing - Benchmark tool compares model performance

Priority: Could Have **Story Points:** 13 **Related Concepts:** AI/ML, Object Detection

---

1.3.4 2.4 Morgan (Manager)

1.3.4.1 Story 13: Performance Dashboard (Must Have) **As** a manager, **I want** a high-level dashboard showing throughput, uptime, and error rates, **So that** I can track KPIs and report to stakeholders.

Acceptance Criteria: - Dashboard shows: picks/hour, uptime %, error rate, cycle time (avg/p95) - Historical trends (daily, weekly, monthly) - Export reports as PDF or CSV - Alerts when KPIs fall below thresholds

Priority: Must Have **Story Points:** 13 **Related Concepts:** Monitoring, Analytics

---

1.3.4.2 Story 14: ROI Calculator (Should Have)

As a manager, **I want** to see a cost-benefit analysis (labor saved vs system cost), **So that** I can justify the investment.

Acceptance Criteria: - Input: labor cost/hour, picks/day (manual), system cost - Output: payback period, NPV, IRR - Compare scenarios (1 robot vs 2 robots, etc.) - Interactive what-if analysis

Priority: Should Have **Story Points:** 8 **Related Concepts:** Business Case, ROI

1.3.4.3 Story 15: Predictive Maintenance (Could Have)

As a manager, **I want** alerts when components (motors, gripper) are predicted to fail, **So that** I can schedule maintenance proactively.

Acceptance Criteria: - Monitor motor temperatures, cycle counts, vibration - ML model predicts remaining useful life (RUL) - Alert when RUL < 2 weeks - Maintenance schedule auto-generated

Priority: Could Have **Story Points:** 21 **Related Concepts:** Predictive Maintenance, AI/ML

1.3.5 2.5 Casey (Maintenance)

1.3.5.1 Story 16: Diagnostic Tools (Must Have)

As a maintenance technician, **I want** a health check tool that tests all sensors and actuators, **So that** I can quickly diagnose faults.

Acceptance Criteria: - One-click health check runs in <5 minutes - Tests: camera (image quality), encoders (position accuracy), F/T sensor (calibration), motors (torque) - Pass/fail report with detailed errors - Guidance for replacing failed components

Priority: Must Have **Story Points:** 13 **Related Concepts:** Testing, Diagnostics

---

1.3.5.2 Story 17: Maintenance Schedule (Should Have)

As a maintenance technician, **I want** a calendar showing upcoming maintenance tasks (e.g., “Lubricate joints every 1000 hours”), **So that** I can plan preventive maintenance.

Acceptance Criteria: - Maintenance tasks defined per component - Calendar view shows next due date - Email/SMS reminders 1 week before due - Log completed tasks (date, technician, notes)

Priority: Should Have **Story Points:** 8 **Related Concepts:** Maintenance Planning

---

1.3.5.3 Story 18: Spare Parts Inventory (Could Have) **As** a maintenance technician, **I want** the system to track spare parts usage and reorder when low, **So that** I don't run out of critical components.

Acceptance Criteria: - Inventory database (part name, quantity, reorder level) - Alert when quantity < reorder level - Integration with procurement system (auto-generate PO) - Track part usage per robot

Priority: Could Have **Story Points:** 13 **Related Concepts:** Inventory Management

---

1.3.6 2.6 Taylor (Data Scientist)

1.3.6.1 Story 19: Data Collection Pipeline (Must Have) **As** a data scientist, **I want** to automatically log images and labels (bounding boxes, poses) for every pick, **So that** I can build a dataset for model retraining.

Acceptance Criteria: - Log RGB-D images, ground-truth poses (from robot FK) - Metadata: timestamp, object class, success/failure - Storage: HDF5 or organized folder structure - Option to anonymize data (blur backgrounds)

Priority: Must Have **Story Points:** 8 **Related Concepts:** Data Management, AI/ML

1.3.6.2 Story 20: Model Retraining Workflow (Should Have) **As** a data scientist, **I want** a Jupyter notebook template for retraining the detection model on new data, **So that** I can improve accuracy without starting from scratch.

Acceptance Criteria: - Template includes: data loading, train/val split, training loop, evaluation - Pre-configured hyperparameters (learning rate, batch size) - MLflow integration for experiment tracking - Export trained model to ONNX/TensorRT

Priority: Should Have **Story Points:** 13 **Related Concepts:** AI/ML, Training Pipeline

---

1.3.6.3 Story 21: A/B Testing for Models (Could Have) **As** a data scientist, **I want** to deploy two models in parallel (A/B test) and compare their performance, **So that** I can validate improvements before full rollout.

Acceptance Criteria: - Route 50% of requests to model A, 50% to model B - Log inference results for both models - Compare metrics: accuracy, latency, error rate - Gradual rollout (10% → 50% → 100%)

Priority: Could Have **Story Points:** 13 **Related Concepts:** AI/ML, Experimentation

---

1.3.7 2.7 Riley (Safety Officer)

1.3.7.1 Story 22: Safety Zone Monitoring (Must Have) **As** a safety officer, **I want** the robot to stop immediately if a human enters the workspace, **So that** we comply with ISO 10218 and prevent injuries.

Acceptance Criteria: - Vision-based human detection (YOLO person class) - Safety-rated laser scanner (optional, for redundancy) - Robot stops within 100ms of detection - Cannot restart until human leaves zone and "Resume" is pressed

Priority: Must Have **Story Points:** 13 **Related Concepts:** Safety, Human-Robot Interaction

---

1.3.7.2 Story 23: Audit Trail (Must Have)

As a safety officer, **I want** immutable logs of all safety events (E-stop, collisions, zone breaches), **So that** I can investigate incidents and ensure compliance.

Acceptance Criteria: - Logs include: timestamp, event type, trigger source, robot state - Logs cannot be edited or deleted (append-only) - Retention: 5 years - Export logs for audits (PDF report)

Priority: Must Have **Story Points:** 8 **Related Concepts:** Logging, Security, Compliance

---

1.3.7.3 Story 24: Force Limiting (Should Have)

As a safety officer, **I want** the robot to limit contact forces to <150N (per ISO/TS 15066), **So that** collaborative operation is safe.

Acceptance Criteria: - F/T sensor monitors contact forces in real-time - Robot stops if force exceeds 150N - Configurable force limits per zone (lower limits near humans) - Monthly force calibration check

Priority: Should Have **Story Points:** 13 **Related Concepts:** Safety, Force Control

---

1.3.8 2.8 Drew (Customer)

1.3.8.1 Story 25: Easy Deployment (Must Have)

As a customer, **I want** the system to be operational within 1 day of delivery, **So that** I minimize downtime.

Acceptance Criteria: - Pre-configured for common use cases (e.g., bin picking, kitting) - Calibration wizard completes in <1 hour - Training materials: video tutorials, quick start guide - Remote support available during setup

Priority: Must Have **Story Points:** 21 **Related Concepts:** Deployment, Documentation

---

1.3.8.2 Story 26: Vendor Support (Must Have)

As a customer, **I want** 24/7 support with <4 hour response time for critical issues, **So that** I can maintain production uptime.

Acceptance Criteria: - Support portal with ticketing system - Phone/email/chat support channels - SLA: critical issues <4 hours, normal issues <24 hours - Remote diagnostics (VPN access with customer approval)

Priority: Must Have **Story Points:** N/A (operational) **Related Concepts:** Support, Maintenance

---

1.3.8.3 Story 27: Performance Guarantee (Should Have)

As a customer, **I want** a guarantee that the system achieves 99% grasp success on my objects, **So that** I trust the system will meet my needs.

Acceptance Criteria: - Pre-deployment testing on customer's objects (100 picks) - Performance report with success rate, cycle time - Money-back guarantee if <99% success after tuning - Quarterly performance reviews

Priority: Should Have **Story Points:** N/A (business process) **Related Concepts:** Performance, SLA

1.4 3. User Stories Mapped to Core Robotics Concepts

Robotics Concept	User Stories
Computer Vision	2, 5, 12, 19, 22
Kinematics (FK/IK)	4, 5, 6, 19
Motion Planning	6, 8, 9
Grasp Planning	7, 12
Control & Execution	1, 4, 10
State Machine	1, 3
Sensor Fusion & Calibration	5, 16
Coordinate Transforms	5, 6
Collision Avoidance	4, 6
Safety & HRI	1, 22, 24
Monitoring & Logging	2, 11, 23
AI/ML	12, 15, 19, 20, 21, 22
Simulation	8
Performance Optimization	10, 13, 27

1.5 4. Prioritization Summary (MoSCoW)

1.5.1 Must Have (17 stories)

Critical for MVP and core functionality: 1, 2, 5, 6, 9, 13, 16, 19, 22, 23, 25, 26

1.5.2 Should Have (10 stories)

Important for usability and production-readiness: 3, 4, 7, 8, 10, 11, 14, 17, 20, 24

1.5.3 Could Have (5 stories)

Nice-to-have, added if time permits: 12, 15, 18, 21, 27

1.5.4 Won't Have (This Release)

Deferred to future versions: - Multi-robot coordination - Cloud-based analytics - Mobile app (iOS/Android)

1.6 5. Epic Grouping

1.6.1 Epic 1: Core Operation

Stories: 1, 2, 3, 4 **Goal:** Enable operators to run the system reliably

1.6.2 Epic 2: System Integration

Stories: 5, 6, 7, 8 **Goal:** Enable integrators to deploy and configure

1.6.3 Epic 3: Developer Experience

Stories: 9, 10, 11, 12 **Goal:** Enable engineers to develop and debug

1.6.4 Epic 4: Business Intelligence

Stories: 13, 14, 15 **Goal:** Provide visibility and ROI to management

1.6.5 Epic 5: Maintenance & Support

Stories: 16, 17, 18 **Goal:** Enable proactive and reactive maintenance

1.6.6 Epic 6: AI/ML Pipeline

Stories: 19, 20, 21 **Goal:** Enable continuous model improvement

1.6.7 Epic 7: Safety & Compliance

Stories: 22, 23, 24 **Goal:** Ensure safe operation and regulatory compliance

1.6.8 Epic 8: Customer Success

Stories: 25, 26, 27 **Goal:** Ensure customer satisfaction and adoption

---

1.7 6. User Story Estimation

1.7.1 Story Points Distribution

- **Small (3-5 points):** 7 stories (Quick wins, 1-2 weeks)
- **Medium (8-13 points):** 13 stories (Standard features, 2-4 weeks)
- **Large (21 points):** 3 stories (Complex features, 4-8 weeks)

1.7.2 Total Effort Estimate

- **Must Have:** ~150 story points (30 weeks with 1 dev, 15 weeks with 2 devs)
- **Should Have:** ~80 story points (16 weeks)
- **Could Have:** ~60 story points (12 weeks)

Estimated MVP Timeline: 6 months (2 developers) for Must Have + Should Have

---

1.8 7. Acceptance Testing Scenarios

1.8.1 Scenario 1: End-to-End Pick-Place (Story 1, 2)

Given 10 objects on a table **When** operator presses “Start” **Then** system picks and places all 10 objects within 30 seconds **And** dashboard shows 100% success rate

1.8.2 Scenario 2: Error Recovery (Story 3)

Given an occluded object (not detectable) **When** system scans the workspace **Then** error message displays “Object not detected - check for occlusions” **And** operator can press “Retry” or “Skip”

1.8.3 Scenario 3: Calibration (Story 5)

Given a new robot installation **When** integrator runs calibration wizard **Then** hand-eye transform computed within 5 minutes **And** validation detects object position with <5mm error

1.8.4 Scenario 4: Safety Stop (Story 22)

Given robot is moving toward pick position **When** human enters safety zone **Then** robot stops within 100ms **And** alarm sounds, “Resume” button required to continue

---

1.9 8. Non-Functional Requirements Derived from Stories

NFR Category	Requirement	Source Stories
Performance	Cycle time 2 sec/object	13, 27
Reliability	Uptime 99.5%	13, 26
Usability	Operator training 4 hours	1, 2, 3
Scalability	Support 1-10 robots on same network	9
Security	Authentication required for config changes	23
Maintainability	Code coverage 80%	9
Portability	Support Ubuntu 22.04, ROS2 Humble	9
Safety	Comply with ISO 10218, ISO/TS 15066	22, 24

1.10 9. Traceability Matrix

User Story	System Requirement	Design Component	Test Case
1	REQ-001: Start/Stop	Task Orchestrator FSM	TC-001: Start button
2	REQ-002: Dashboard	Grafana + Prometheus	TC-002: Dashboard load
5	REQ-005: Calibration	Hand-Eye Calibration Node	TC-005: Calib accuracy
9	REQ-009: Modularity	ROS2 Package Structure	TC-009: Unit tests
22	REQ-022: Safety Zone	Human Detection Node	TC-022: Stop on detect

---

1.11 10. Conclusion

This user story collection provides: - **27 user stories** covering 8 personas - **Mapped to core robotics concepts** (vision, planning, control, safety, AI) - **Prioritized** using MoSCoW (17 Must Have, 10 Should Have, 5 Could Have) - **Grouped into 8 epics** for sprint planning - **Estimated at 290 story points** total (~58 weeks with 1 developer) - **Acceptance criteria** and test scenarios for validation

Next Steps: 1. Refine story points with development team 2. Create detailed tasks for each story (subtasks in Jira/Linear) 3. Sprint planning: allocate stories to 2-week sprints 4. Continuous backlog grooming based on feedback

---

Document Status: Complete **Last Updated:** 2025-10-18 **Author:** Product Management Team **Review Status:** Pending Stakeholder Review