Product Requirements Document

Motion Tracker

Video-Based Motion Analysis for Physics Education

Version 1.0 | January 2026

Author	Tyler [Last Name]
Status	Draft
Target Platform	Web (Primary), iOS/Android (Future)

1. Executive Summary

Motion Tracker is a web application designed to help physics students and educators analyze real-world motion through video. By allowing users to upload videos, track objects frame-by-frame, and automatically generate position, velocity, and acceleration data, the app bridges the gap between theoretical physics concepts and observable phenomena.

The application draws inspiration from established tools like Vernier Video Physics while focusing on modern UX patterns, accessibility, and integration with contemporary educational workflows. A web-first approach ensures immediate cross-platform access for classrooms with mixed devices, faster iteration cycles, and no app store approval delays.

1.1 Problem Statement

Physics education often struggles to connect abstract kinematic equations with real-world motion. Students memorize formulas without developing intuition for how objects actually move. Existing video analysis tools are either expensive (requiring hardware bundles), outdated (designed for desktop-first workflows), platform-locked (iOS or desktop only), or lack the polish expected of modern web applications.

1.2 Solution Overview

Motion Tracker provides an intuitive, web-first experience for analyzing motion. Students can upload a video of a ball being thrown, a car rolling down a ramp, or any moving object, then trace its path frame-by-frame. The app automatically calculates and visualizes position, velocity, and acceleration, making abstract concepts tangible and explorable. The web-first approach means students can access the tool from any device with a browser—Chromebooks, iPads, laptops, or phones—without installation.

2. Target Users

2.1 Primary Personas

High School Physics Student (Ages 14-18): Taking introductory or AP Physics. Needs to complete lab assignments involving motion analysis. Comfortable with smartphones but may have limited physics background. Wants quick, intuitive workflows that produce results they can include in lab reports.

Physics Educator: Teaching at high school or introductory college level. Needs a tool that works reliably across student devices. Wants to assign video analysis labs without extensive setup. Values data export capabilities for grading and analysis. May need to demonstrate concepts in real-time during class.

2.2 Secondary Personas

College Physics Student: Taking introductory mechanics courses. May need more advanced analysis features. Often working on tight lab deadlines.

Sports Coach or Athlete: Analyzing athletic motion (golf swing, pitch mechanics, running form). Interested in velocity and acceleration patterns. Not necessarily physics-trained but understands basic motion concepts.

Curious Hobbyist: Anyone interested in understanding the physics of everyday motion. May want to analyze YouTube videos or personal recordings.

3. User Stories

3.1 Core User Stories (MVP)

1. **As a student,** I want to upload a video from my camera roll so that I can analyze motion I've already recorded.
2. **As a student,** I want to record video directly in the app so that I can capture motion in real-time during lab.
3. **As a student,** I want to step through video frame-by-frame so that I can precisely mark object positions.
4. **As a student,** I want to set a scale reference (known distance) so that pixel measurements convert to real-world units.
5. **As a student,** I want to tap on the object in each frame to mark its position so that the app can track its motion.
6. **As a student,** I want to see the tracked path overlaid on the video so that I can visualize the trajectory.
7. **As a student,** I want to view graphs of position vs. time (X and Y separately) so that I can analyze the motion quantitatively.
8. **As a student,** I want to view a data table with time, X position, and Y position so that I can see the raw measurements.
9. **As a teacher,** I want students to be able to export data as CSV so that they can perform additional analysis in spreadsheets.
10. **As a teacher,** I want students to be able to save and reopen their analysis projects so that work isn't lost between sessions.

3.2 Enhanced User Stories (Post-MVP)

1. **As a student,** I want to see velocity vectors displayed on the video so that I can visualize speed and direction.
2. **As a student,** I want to view velocity vs. time and acceleration vs. time graphs so that I can analyze all kinematic quantities.
3. **As a student,** I want the app to automatically track the object after I mark it initially so that manual frame-by-frame marking isn't required.
4. **As a student,** I want to track multiple objects simultaneously so that I can compare their motions.
5. **As a student,** I want to add a best-fit line or curve to my graphs so that I can determine equations of motion.
6. **As a teacher,** I want to create and share pre-configured video templates so that students have consistent starting points for labs.

4. Functional Requirements

4.1 Video Input

- Support video upload from device photo library (MOV, MP4, HEVC)
- In-app video recording with standard camera controls
- Video trimming to select analysis region
- Support frame rates up to 240fps for slow-motion analysis
- Automatic extraction of video metadata (frame rate, duration, resolution)

4.2 Coordinate System Setup

- Scale calibration: User marks two points of known distance
- Origin placement: User taps to set coordinate system origin
- Axis orientation: User defines positive X and Y directions
- Support for tilted coordinate systems (rotated axes)

4.3 Object Tracking

- Manual point marking: User taps object center in each frame
- Undo/redo for point placement
- Visual feedback showing tracked path overlay
- Point adjustment: Drag to reposition misplaced points
- Frame skip option (analyze every Nth frame)

4.4 Data Analysis and Visualization

- Position vs. time graphs (separate X(t) and Y(t) plots)
- Y vs. X trajectory plot
- Interactive graphs with pinch-to-zoom and pan
- Data point highlighting: Tap graph point to jump to video frame
- Scrollable data table with time, X, Y columns
- Calculated columns: velocity (vx, vy), speed, acceleration (ax, ay)

4.5 Video Playback Controls

- Play/pause
- Frame-by-frame advance (forward and backward)
- Scrubber for direct navigation
- Playback speed control (0.25x to 2x)
- Frame counter and timestamp display

4.6 Project Management

- Save projects locally with all settings and tracked data
- Project list with thumbnails and metadata
- Delete and rename projects
- Auto-save during analysis

4.7 Data Export

- CSV export of data table
- Image export of graphs
- Share via standard iOS share sheet
- Copy data to clipboard

5. Technical Specifications

5.1 Platform and Technology Stack

Component	Technology
Platform	Web (Modern browsers: Chrome, Safari, Firefox, Edge)
Frontend Framework	React 18+ with TypeScript
Styling	Tailwind CSS + shadcn/ui components
Video Processing	HTML5 Video API + Canvas API for frame extraction
Charting	Recharts or D3.js for interactive graphs
State Management	Zustand or React Context + useReducer
Backend	Node.js/Fastify on Railway
Database	Supabase (PostgreSQL + Auth + Storage)
Local Storage	IndexedDB (via Dexie.js) for offline project caching
Future: Auto-Track	TensorFlow.js or MediaPipe for browser-based object tracking
Future: Mobile Apps	React Native or PWA with native wrappers

5.2 Video Handling in Browser

Browser-based video analysis requires careful handling of frame extraction. The HTML5 Video element combined with Canvas provides the foundation for this functionality.

Frame Extraction: Use video.currentTime to seek to specific frames, then draw to canvas with drawImage(). For precise frame-by-frame control, calculate frame times as frameNumber / frameRate.

Supported Formats: MP4 (H.264) has universal browser support. WebM (VP9) as fallback. MOV files may need transcoding for non-Safari browsers.

Performance Considerations: Large videos should be processed client-side to avoid upload bandwidth. Consider Web Workers for heavy calculations. Use requestAnimationFrame for smooth playback.

Touch/Mouse Handling: Pointer Events API for unified touch and mouse input. Implement pinch-to-zoom via touch events for precise point placement on mobile browsers.

5.3 Data Model

Project

- id: string (UUID)
- name: string
- createdAt: Date (ISO 8601)
- modifiedAt: Date
- videoUrl: string (Supabase Storage URL or local blob URL)
- frameRate: number
- duration: number (seconds)
- videoWidth: number
- videoHeight: number
- thumbnailUrl: string

CoordinateSystem

- origin: { x: number, y: number } (pixel coordinates)
- scalePoint1: { x: number, y: number }
- scalePoint2: { x: number, y: number }
- scaleDistance: number (real-world units)
- unit: string ("m", "cm", "ft", etc.)
- rotationAngle: number (radians)

TrackedObject

- id: string (UUID)
- name: string
- color: string (hex color code)
- dataPoints: DataPoint[]

DataPoint

- frameNumber: number
- time: number (seconds)
- pixelPosition: { x: number, y: number }
- worldPosition: { x: number, y: number } (calculated from scale)
- isManuallyPlaced: boolean

5.4 Key Calculations

Pixel to World Conversion: worldPosition = (pixelPosition - origin) * (scaleDistance / pixelDistance)

Velocity: v = (position[n+1] - position[n-1]) / (time[n+1] - time[n-1]) using central difference

Acceleration: a = (velocity[n+1] - velocity[n-1]) / (time[n+1] - time[n-1]) using central difference

Speed: speed = sqrt(vx^2 + vy^2)

6. UI/UX Requirements

6.1 Design Principles

1. **Focused Workflow:** Guide users through a clear sequence: import → calibrate → track → analyze → export
2. **Touch-First:** All interactions optimized for finger input; precision tracking via zoom
3. **Immediate Feedback:** Every tap shows visual confirmation; tracked path updates in real-time
4. **Forgiving:** Easy undo, adjustable points, non-destructive edits

6.2 Screen Flow

1. **Home/Project List:** Grid of project thumbnails with create new option
2. **Video Import:** Camera roll picker or in-app recording
3. **Calibration:** Set scale (mark known distance) and origin placement
4. **Tracking:** Main analysis view with video, frame controls, and point marking
5. **Analysis:** Split view with graphs and data table
6. **Export:** Options for CSV, images, sharing

6.3 Key UI Components

Video Canvas: Full video frame with overlay layer for tracked points, path trails, and velocity vectors. Support pinch-to-zoom for precise point placement.

Frame Scrubber: Timeline showing frame thumbnails; drag to navigate. Current frame indicator. Frame counter display (e.g., "Frame 47/62").

Graph Panel: Resizable panel that can be expanded or minimized. Toggle between different graph types. Interactive: tap points to sync with video.

Data Table: Scrollable table synced with video position. Editable cells for manual correction. Column sorting.

7. MVP Scope Definition

7.1 Core Motion Analysis (MVP)

- Video upload from local device (MP4, WebM)
- Frame-by-frame navigation with keyboard shortcuts
- Scale calibration (two-point reference with unit selection)
- Origin and axis definition (including tilted axes for ramp experiments)
- Manual point tracking (single object)
- Path overlay with adjustable trail length
- Responsive design (desktop + tablet browsers)

7.2 Graphing and Analysis (MVP)

Based on AP Physics 1 lab requirements, the following graph types are essential for students to analyze motion, verify relationships, and perform linearization techniques:

- Position vs. time graphs (X(t) and Y(t) separately)
- Velocity vs. time graphs (Vx(t) and Vy(t)) — essential for verifying constant acceleration
- Y vs. X trajectory plot
- Linear regression with slope and y-intercept display
- Customizable axis variables (plot any column vs. any other column)
- Calculated columns: velocity, speed, acceleration
- Derived columns (t^2, x^2, etc.) for linearization of non-linear relationships
- Interactive graph-video sync (click point to jump to frame)

7.3 Data Management (MVP)

- Cloud storage via Supabase (projects persist across devices)
- User accounts with email/password and Google OAuth
- Guest mode with local-only storage for quick demos
- CSV export of data table
- PNG export of graphs
- Project sharing via link (view-only for collaboration)

7.4 Deferred to Post-MVP

- Automatic object tracking (TensorFlow.js)
- Multiple object tracking
- Velocity vectors overlay on video

- Quadratic and polynomial curve fitting
- Uncertainty/error analysis tools
- Stroboscopic composite image generation
- In-browser video recording via webcam
- Angle measurement tool for rotational motion
- Free-body diagram drawing tool
- Native iOS and Android apps

8. Development Roadmap

Phase 1: Foundation (Weeks 1-4)

- Project scaffolding: React + TypeScript + Vite
- Supabase setup: Auth, Database schema, Storage buckets
- Video upload and HTML5 Video playback
- Canvas-based frame extraction and display
- Frame-by-frame navigation with keyboard shortcuts

Phase 2: Core Tracking (Weeks 5-8)

- Calibration flow: scale reference with unit selection
- Origin and tilted axis placement (for ramp experiments)
- Point marking with Pointer Events API
- Canvas overlay for path visualization
- Undo/redo with state history

Phase 3: Analysis (Weeks 9-12)

- Position-time and velocity-time graphs with Recharts
- Linear regression with slope/intercept display
- Custom axis selection (any column vs any column)
- Calculated and derived columns (v, a, t^2, x^2)
- Interactive data table with edit capability
- Graph-to-video sync (click point to jump to frame)

Phase 4: Storage and Export (Weeks 13-16)

- Cloud project persistence via Supabase
- Project list with thumbnails and search
- CSV and PNG export
- Project sharing via link
- Responsive design polish
- Deploy to Railway, public beta launch

Phase 5: Classroom Features (Post-MVP)

- Teacher accounts with class management
- Assignment creation and distribution
- Student submission workflow

- Pre-built lab templates (AP Physics 1 investigations)
- Basic gradebook with CSV export

Phase 6: Advanced Analysis (Future)

- Automatic object tracking with TensorFlow.js
- Multiple object tracking
- Velocity vectors overlay
- Polynomial curve fitting
- Uncertainty analysis tools

Phase 7: Mobile and Platform Expansion (Future)

- PWA with offline support
- React Native apps for iOS and Android
- Native camera integration
- LMS integrations (Canvas, Google Classroom, Schoology)

9. Success Metrics

9.1 User Engagement

- Projects created per user
- Average points tracked per project
- Data exports per project
- Session duration

9.2 Quality Indicators

- App Store rating (target: 4.5+)
- Crash-free sessions (target: 99%+)
- User retention (7-day, 30-day)

9.3 Educational Impact (Qualitative)

- Teacher testimonials
- Adoption in physics courses
- Feature requests from educators

10. Future Considerations

10.1 Potential Features

- **Automatic Object Tracking:** Use TensorFlow.js or MediaPipe for browser-based ML tracking after initial object selection
- **Stroboscopic View:** Composite image showing object at multiple time points
- **Angle Measurement:** Measure angles between points for rotational motion analysis
- **Pre-Built Labs:** Template projects with guided instructions for common physics experiments
- **Integration with LMS:** Export directly to Google Classroom, Canvas, or other learning management systems
- **Collaborative Analysis:** Real-time collaboration on projects, similar to Figma
- **Video URL Import:** Analyze videos from YouTube or other sources via URL (with appropriate licensing)

10.2 Mobile App Strategy

- **PWA First:** Progressive Web App with offline support covers most mobile use cases without app store distribution
- **React Native:** If native apps needed, share business logic and state management with web codebase
- **Native Camera:** Primary advantage of native apps—direct slow-motion recording integration

10.3 Monetization Options

- **Freemium:** Basic tracking free; advanced features (auto-track, multiple objects, cloud storage) require subscription
- **Education Pricing:** School/district volume licensing with teacher admin dashboard
- **Open Core:** Core analysis tool free and open source; cloud features and LMS integration paid

11. Long-Term Vision: K-12 STEM Learning Platform

Motion Tracker serves as the foundation for a broader K-12 math and science learning management system. The motion analysis tool establishes credibility with physics teachers, and the platform expands to serve the full STEM curriculum from elementary through high school.

11.1 Platform Evolution

Phase 1 — Motion Analysis Tool (Year 1): Launch Motion Tracker as a standalone web app. Build user base among AP Physics teachers. Establish reputation for quality and ease of use. Gather feedback for LMS features.

Phase 2 — Classroom Features (Year 2): Add teacher accounts with class management. Assignment creation and distribution. Student submissions and teacher feedback. Basic gradebook integration.

Phase 3 — Expanded Science Tools (Year 2-3): Add complementary analysis tools: data logger integration for sensors, graphing calculator replacement, simulation library (PhET-style), lab report builder with templates.

Phase 4 — Full LMS (Year 3+): Curriculum-aligned content library. Standards mapping (NGSS, Common Core Math). Progress tracking and analytics. Parent communication portal. District-level administration.

11.2 LMS Core Features

For Teachers

- Class roster management with student accounts
- Assignment builder: attach videos, set due dates, rubrics
- Pre-built lab templates aligned to AP Physics 1 investigations
- Submission review with inline feedback on graphs and data
- Gradebook with CSV export for SIS integration
- Analytics: identify struggling students, track skill mastery

For Students

- Dashboard showing assignments and due dates
- Guided lab workflows with step-by-step instructions
- Lab notebook / portfolio for storing artifacts
- Progress tracking toward learning objectives
- Peer collaboration tools for group labs

For Administrators

- District-wide deployment and user provisioning
- Usage analytics across schools
- Standards alignment reporting
- SSO integration (Clever, ClassLink, Google Workspace for Education)

11.3 Subject Expansion Roadmap

Subject	Tools	Grade Levels
Physics	Motion Tracker, circuit simulator, wave analyzer	9-12, AP
Chemistry	Molecular modeling, reaction balancer, titration sim	9-12, AP
Biology	Data collection, population modeling, genetics sim	6-12, AP
Math	Graphing calculator, geometry tools, statistics	K-12
Earth Science	Weather data analysis, plate tectonics sim	6-9
Elementary STEM	Simplified motion tracker, measurement tools	K-5

11.4 Competitive Positioning

The K-12 STEM edtech space includes established players (Vernier, PASCO, PhET) and general LMS platforms (Canvas, Schoology, Google Classroom). Our differentiation:

- **Purpose-built for STEM:** Unlike general LMS platforms, every feature is designed for science and math instruction
- **Modern tech stack:** Web-first, mobile-friendly, no desktop software installation required
- **No hardware lock-in:** Works with any device, unlike Vernier/PASCO ecosystems that require proprietary sensors
- **Teacher-designed:** Built by someone with 12 years of physics classroom experience
- **Affordable:** Freemium model makes it accessible to underfunded schools

Appendix A: Competitive Analysis

App	Strengths	Weaknesses	Price
Vernier Video Physics	Established brand, comprehensive features, velocity vectors	Dated UI, requires Vernier ecosystem	$4.99
Tracker (Desktop)	Free, powerful analysis, auto-tracking	Desktop only, steep learning curve, Java-based	Free
Physics Toolbox	Sensor suite, modern UI	Limited video analysis	Free/Paid

Appendix B: Glossary

Frame Rate: Number of individual frames captured per second (fps). Higher frame rates allow for more precise motion analysis.

Scale Calibration: The process of establishing the relationship between pixel distances in the video and real-world distances.

Central Difference Method: A numerical technique for calculating derivatives that uses data points on both sides of the target point, providing better accuracy than forward or backward differences.

Projectile Motion: Motion of an object thrown or projected into the air, subject only to gravity and air resistance.

Kinematics: The branch of mechanics that describes the motion of objects without considering the forces that cause the motion.