

JPL - MoonTrek Augmented Reality User Manual

Group 2

December 2025

Jira Project Link

<https://your-team-domain.atlassian.net/jira/software/projects/MTAR>

Project Key: MTAR

Formal Objective Breakdown

Snapshot 1: Foundation & Core Infrastructure

Set up Vue.js frontend and Express.js backend, implement MySQL database for user and image data, create image upload functionality with metadata extraction (EXIF data), integrate NASA MoonTrek API, and complete initial documentation (SDD, SRS).

Snapshot 2: 3D Model Generation & Image Registration

Develop Three.js-based 3D model of Earth-Moon-Sun system using WebGL, implement SIFT algorithm for context-aware image registration, create reference image generation from 3D model, and build coordinate mapping system from WAC composite to user images.

Snapshot 3: Overlay System & Circle Detection

Implement circle detection algorithm to identify moon in user images, build automatic annotation overlays for lunar features using MoonTrek API data, add points of interest (craters, maria, landing sites), and create transformation matrix for accurate overlay positioning.

Snapshot 4: Final Polish & YourMoon Database

Complete YourMoon image database with cropping interface, implement machine learning validation for moon images, optimize image registration performance for real-time telescope feeds, and document future enhancements.

Goals

Provide amateur astronomers with automatic annotation of Moon telescope images through a web-based interface. Using context-aware image registration with SIFT algorithm and 3D Earth-Moon-Sun modeling via Three.js, the application overlays lunar feature names, craters, maria, and landing sites from NASA's MoonTrek database onto user-uploaded images.

Why it matters: Bridges the gap between professional NASA lunar data and amateur astronomers, enhances astronomical education, supports citizen science, and eliminates the need to manually cross-reference lunar atlases while observing.

How to Download and Access

Accessing the Website

MoonTrek AR is a web-based application built with Vue.js - no downloads or installations required!

Simply visit: <http://moontrek-ar.com>

Compatible with:

- Desktop browsers with WebGL support (Chrome, Firefox, Safari, Edge)
- Mobile browsers (iOS Safari, Android Chrome)
- Tablet devices

Using the Application

Step 1: Visit the Website

Navigate to <http://moontrek-ar.com> on any device with internet access and WebGL-capable browser.

Step 2: Upload Your Moon Image

- Click "Upload Image" or drag and drop your telescope image
- Supported formats: JPEG, PNG (max 10MB)
- Image will be processed with circle detection to identify the moon
- Metadata (timestamp, GPS coordinates) will be extracted automatically
- If metadata is missing, you'll be prompted to enter location and time manually

Step 3: View 3D Model and Registration

- Processing takes 30-60 seconds
- Three.js generates photorealistic 3D model of Earth-Moon-Sun system
- Reference image is created based on your location and timestamp
- SIFT algorithm performs context-aware image registration
- View transformation matrix results

Step 4: Explore Annotations

- See automatic overlays with lunar feature names
- View craters, maria (lunar seas), and landing sites
- Access MoonTrek API data for points of interest
- Use dropdown menu to toggle different data layers
- Adjust overlay opacity for better visualization

Optional: Contribute to YourMoon Database

Create a free account to contribute your moon images to the YourMoon database:

- Upload and crop moon images
- Images validated through machine learning
- Help build test database for registration accuracy
- Access your upload history

Features

- **Context-Aware Registration:** SIFT algorithm with 3D model reference images
- **Real-time 3D Modeling:** Three.js WebGL rendering of planetary positions
- **Circle Detection:** Automatic moon identification in images
- **NASA Data Integration:** Direct access to MoonTrek API overlays
- **Metadata Extraction:** Automatic timestamp and GPS coordinate parsing
- **Coordinate Mapping:** One-to-one correspondence from WAC composite to user image

Technical Stack

Frontend: Vue.js with WebGL support for Three.js 3D rendering

Backend: Express.js (Node.js) with image processing routes

Database: MySQL for user data and moon image metadata

Image Processing: Python with SIFT algorithm (OpenCV)

3D Graphics: Three.js for Earth-Moon-Sun modeling with texture mapping

APIs: NASA MoonTrek API for lunar feature data