PathArt – GPS Art Route Generator
Web Application Development

Introduction

PathArt is a creative web-based system that converts user sketches into real-world GPS-art routes. The application integrates multiple technical domains including vectorisation, geospatial mapping, route generation, and geo-format conversion.
This document presents the research underpinning the development of PathArt, including tools, technologies, libraries, routing systems, and related work.

Background & Context

GPS Art

GPS art is the practice of creating images by walking, cycling, or running routes that form a shape when viewed on a map. It gained popularity through platforms such as **Strava**, **Garmin**, and **Komoot**, where athletes share creative route shapes.
Traditional GPS art is manually planned, requiring:

- Manual tracing over digital maps
- Trial/error to follow roads
- External tools to check routability

PathArt automates this process.

The Need for Automation

Most artists rely on:

- Paper sketches
- Manually tracing them on Google Maps
- Exporting GPX files using third-party tools

These steps are slow, error-prone, and require technical knowledge.
PathArt removes this complexity.

Key Research Areas

Image Vectorisation

To convert user sketches into clean outlines, two JavaScript libraries were investigated:
Potrace.js

- Browser port of the classic Potrace algorithm
- Converts raster images into scalable vector paths
- Handles thresholding, smoothing, and noise removal
- Suitable for simple outlines

Paper.js

- Vector graphics framework for the web
- Provides Bézier curve simplification and manipulation
- Useful for cleaning wobbly lines and reducing point counts

Chosen Approach: Potrace.js for initial vectorisation + optional Paper.js for simplification.

Mapping Librarie

Two main technologies were evaluated:

Leaflet.js

- Lightweight, open-source
- Easy to overlay SVG paths
- Works with OpenStreetMap tiles (free)

Mapbox GL JS

- More advanced rendering (WebGL)
- Smoother animations
- But requires paid usage for high traffic

Chosen Approach: Leaflet.js due to simplicity and cost-free usage.

Routing Engines

To snap vector outlines to real-world roads, navigation APIs are required:

OpenRouteService (ORS)

- Free tier: ~2,000 requests/day
- Self-hostable on Docker
- Supports profiles (walk/run/cycle)
- Provides detailed GeoJSON output

Mapbox Directions API

- Easy to integrate
- High-quality routing
- Usage-based pricing

Chosen Approach: ORS for development; Mapbox optional for performance testing.

Geo Formats Research

Three main formats were examined:

1. GeoJSON

- Ideal for working with geographic line data
- Human-readable
- Used within the app for routing and editing

2. GPX

- XML-based route format used by Strava/Komoot
- Requires conversion from GeoJSON
- Generated using togpx or backend libraries (gpxpy)

3. SVG

- Useful for representing vectorised drawings
- Not suitable for GPS routing directly

Related Work & Academic References

GPS Art Analysis (Zhao et al., 2020)

- Discusses automated generation of artistic GPS paths
- Highlights challenges in following street networks
- Supports importance of simplification & map fitting

OpenStreetMap (OSM)

- Open mapping data used by Leaflet & ORS
- Community-driven, accurate urban layout data

Strava API

- Demonstrates GPX import expectations
- Confirms compatibility requirements (track format)

Technology Evaluation

Backend Technologies

Python Flask

- Lightweight
- Simple routing endpoints
- Easy integration with ORS
- Ideal for academic projects

Node.js + Express

- Fast API routing
- NPM ecosystem
- Good choice for JavaScript full-stack

Chosen Approach: Flask (clearer for a Python-based workflow).

Risks & Challenges

R1 — Routing API Limits

Free API tiers may restrict heavy usage.

R2 — Unroutable Shapes

Sketches might cross:

- Rivers
- Buildings
- Motorways
- Dead-end streets

Mitigation: Provide warnings or auto-adjust points.

R3 — Performance Constraints

Large drawings may produce too many sampling points → slow routing.

Mitigation: Adaptive simplification.

R4 — Vectorisation Errors

Poor-quality sketches may produce broken outlines.

Mitigation: Pre-processing filters + threshold controls.

Future Enhancements

AI Sketch Enhancement

Apply ML techniques to:

- Smooth curves
- Symmetrise shapes
- Repair gaps
- Auto-clean user sketches

Smart Area Suggestion

System recommends the best neighbourhood for the artwork based on:

- Street density
- Curviness
- Grid alignment

Collaborative GPS Art

Multiple users generate different parts of the same artwork.

Conclusion

This research confirms that PathArt is technically feasible using existing open-source tools and APIs.

The combination of:

- client-side vectorisation
- Leaflet-based mapping
- ORS routing
- GPX export

creates a complete and efficient workflow for turning sketches into GPS art.

The system is flexible, extendable, and supports numerous future enhancements.