MCP SERVER | NO CODE | CLOUD HOSTED

Google Roads MCP

Map raw GPS data to accurate, actionable street geometry.

Google Roads provides precise mapping tools to match raw GPS data to actual road networks anywhere in the world. Use this MCP to snap scattered coordinates to the nearest roads, reconstruct accurate travel paths, and retrieve official speed limit data for specific segments. It's essential infrastructure for anyone building location-aware applications or analyzing vehicle telemetry.

A+ Quality Score 100/100

gps-tracking | map-matching | road-geometry | speed-limits | telemetry

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/google-roads

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

Google Roads MCP

4 tools available
Cloud-hosted on Vinkius

This MCP lets your AI client take control of complex geospatial tasks that usually require dedicated GIS software. Instead of struggling with noisy GPS feeds or guessing where a track went, you can pass raw coordinate data and get clean road geometries back. You'll find tools to snap entire tracks to the most likely roads traveled, identifying every point along the way. It also helps you figure out which major roads are near individual points, even if they aren't on a path. Plus, it retrieves posted speed limits for those identified segments. Because Vinkius hosts this MCP, your agent can access all these advanced mapping capabilities—from snapping paths to getting specific place IDs—all through one conversation.

Core Capabilities

01 — Match GPS tracks to roads

Transforms a series of raw coordinates into a clean, continuous path that follows the actual road geometry.

02 — Find nearest roadside segment

Identifies the specific, closest road segment for individual GPS points, treating each point independently rather than as part of a path.

03 — Retrieve speed limit data

Gets posted legal speed limits (in km/h) by referencing the unique identifiers of matched road segments.

04 — Combine snapping and speed checks

Performs both map matching and speed limit retrieval in a single, efficient request to save API calls.

05 — Correct noisy GPS data

Cleans up messy telemetry from vehicles by converting scattered points into accurate road-level positions.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/google-roads** — connect your AI agent in three steps.

01 Subscribe to this MCP and provide your Google Maps Platform API key, ensuring the Roads API is enabled.

02 Your AI client sends raw GPS coordinates or a sequence of points it needs mapped against actual roads.

03 The MCP processes the data, returning snapped road geometries, associated place IDs, and optional speed limit values.

The bottom line is that you stop worrying about API keys and complex parameters; your agent just asks for what it needs—a clean map or a speed check—and gets a structured answer.

Built For

If you deal with vehicle telemetry, field mapping, or geospatial data analysis daily, this MCP is required. It's built for the developer tired of cleaning up noisy GPS logs and the analyst who needs reliable road context for compliance checks.

Fleet Manager

Needs to process raw vehicle telemetry to identify traveled roads, visualize accurate routes, and check if speed limits were exceeded.

Mapping Developer

Uses the MCP to convert rough GPS points into clean road geometries for visualization in custom mapping applications or games.

GIS Analyst

Snaps scattered point clusters to known road networks for spatial analysis, ensuring data integrity when integrating with other geographical datasets.

What Changes When You Connect

01 Accurate Route Reconstruction: Use `snap_to_roads` to turn messy, noisy vehicle tracks into clean, continuous road geometries for visualization and historical analysis.

02 Efficiency Gains with Combined Calls: The `get_snapped_speed_limits` tool eliminates the need for multiple API calls; it gives you both the snapped location and the speed limit in one request.

03 Targeted Point Analysis: When you only have scattered points, use `get_nearest_roads`. This function treats each coordinate independently to find the closest road segment without assuming a path exists between them.

04 Compliance Monitoring: By pairing `snap_to_roads` with `get_speed_limits`, you can programmatically check if recorded speeds match posted legal limits for specific segments.

05 Data Foundation: The resulting place IDs are reusable. You can take the output from any snapping tool and feed it directly into other mapping services to enrich your data.

Real-World Applications

Analyzing Driver Compliance

A fleet safety analyst needs to audit a driver's route. Instead of manually cross-referencing GPS logs with local maps, the agent uses `snap_to_roads` to map the path and then feeds those resulting place IDs into `get_speed_limits`. This instantly generates a report showing every segment where the recorded speed violated the posted limit.

Mapping Field Data Collection

A GIS professional collects GPS points across an area but doesn't have a continuous path. They use `get_nearest_roads` to snap each individual point cluster to its nearest road segment, allowing them to map the entire collection accurately for later spatial analysis.

Building Real-Time Vehicle Dashboards

A developer needs a dashboard that shows both a vehicle's current location and the legal speed limit. They use `get_snapped_speed_limits` to get this combined data in one API call, ensuring the visualization is always accurate and up-to-date.

Reconstructing Historical Routes

A mapping developer needs to clean up years of archived GPS telemetry from a vehicle. They pass the raw data into `snap_to_roads` which smooths out all the signal noise and reconstructs the true, navigable road path for accurate visualization.

Patterns to Avoid

Treating GPS points as a solid line

✗ AVOID

Running `snap_to_roads` on 10 independent coordinates that were actually taken far apart. The tool tries to connect them into one continuous, likely inaccurate path.

✓ INSTEAD

If the points are truly scattered and not part of a single journey, use `get_nearest_roads`. This correctly matches each point individually without assuming connectivity.

Over-calling speed limits

✗ AVOID

Calling `snap_to_roads` to get place IDs, then making a separate call using those place IDs for the speed limit. This adds unnecessary latency and consumes more API credits.

✓ INSTEAD

Use `get_snapped_speed_limits`. It combines both functions into one request, saving time and resources.

Ignoring point type

✗ AVOID

Using any road matching tool when the coordinates are known to be from a stationary object (like a pole or building corner), not a vehicle.

✓ INSTEAD

Review your data first. If you need the nearest road for discrete, non-sequential points, `get_nearest_roads` is the specific function designed for that use case.

The Right Fit

Use this MCP if your primary problem involves taking raw GPS coordinates and needing to know exactly what physical road segment they correspond to, or what speed limits apply on that road. This tool is specialized for map matching and telemetry analysis.

Do NOT use it if you simply need general geographic lookups (like finding the nearest gas station) or if your data is already in a perfectly clean format. For simple point-to-POI searches, other location services are better suited. If your goal is purely to validate coordinates against an indexed dataset of known locations, check out tools designed for database validation instead.

The frustration of raw GPS data

Every time you pull a log file from a vehicle or run a survey that uses GPS coordinates, you get thousands of numbers. But those numbers are just points floating in space; they don't tell you if the point is actually on a drivable road, what speed limit applies there, or how to draw a clean path between them. You spend hours cleaning up noise and guessing at geometry.

With this MCP, your AI agent handles that entire process. It takes those raw numbers and reliably converts them into usable road geometries, identifying the exact segment and associated place ID for every point you give it. Your output isn't just data; it’s a validated map.

Getting speed limit context with Google Roads

Before this, figuring out the legal speed on a segment required multiple steps: first finding the road geometry, then taking the resulting ID to another service just for the limits. This was slow, complex, and prone to breaking if any step failed.

Now, you execute `get_snapped_speed_limits`. It handles both the mapping and the legal data retrieval in a single pass. The difference is simple: you get actionable compliance reports without building custom multi-step API wrappers.

Google Roads: 4 Tools for Geospatial Analysis

These tools allow you to snap raw GPS coordinates to actual road networks, find nearest segments, reconstruct paths, and pull legal speed limit data.

#	TOOL	DESCRIPTION
01	`get_nearest_roads`	Finds the nearest existing road segment and its unique ID for up to 100 individual GPS coordinates, treating each point separately.
02	`snap_to_roads`	Matches a continuous sequence of GPS points (up to 100) to the most likely road path and generates smoothed geometry with unique IDs.
03	`get_snapped_speed_limits`	Performs both road snapping and speed limit retrieval simultaneously, providing matched coordinates and current posted limits in one call.
04	`get_speed_limits`	Retrieves the specific legal speed limits (km/h) for any known road segment using its place ID.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U Snap these GPS coordinates to roads: 40.7128,-74.0060|40.7135,-74.0055|40.7142,-74.0048

Snapping GPS coordinates to roads... 3 points snapped successfully. Point 1: Original (40.712800, -74.006000) → Snapped to (40.712820, -74.005980), placeId: ChIJd8BlQ2BZwokRAFUEcm_qrcA. Point 2: Original (40.713500, -74.005500) → Snapped to (40.713510, -74.005490), placeId: ChIJd8BlQ2BZwokRAFUEcm_qrcB. Point 3: Original (40.714200, -74.004800) → Snapped to (40.714190, -74.004810), placeId: ChIJd8BlQ2BZwokRAFUEcm_qrcC. All points snapped to Broadway in Lower Manhattan. Would you like speed limit data for these road segments?

U Get speed limits for these place IDs: ChIJd8BlQ2BZwokRAFUEcm_qrcA|ChIJd8BlQ2BZwokRAFUEcm_qrcB

Fetching speed limits... 2 road segments found. Segment 1: placeId ChIJd8BlQ2BZwokRAFUEcm_qrcA, speed limit 40 km/h (25 mph), road name Broadway, New York NY. Segment 2: placeId ChIJd8BlQ2BZwokRAFUEcm_qrcB, speed limit 40 km/h (25 mph), road name Broadway, New York NY. Both segments are in a downtown urban zone with reduced speed limits.

U Find the nearest road to these coordinates: 34.0522,-118.2437 and 34.0530,-118.2445

Finding nearest roads... Point 1 (34.0522, -118.2437) snapped to Spring Street, Los Angeles CA, placeId: ChIJYWRSrVHHwoARpVI2b3cFzQE. Point 2 (34.0530, -118.2445) snapped to Broadway, Los Angeles CA, placeId: ChIJYWRSrVHHwoARpVI2b3cFzQF. Both points are in downtown Los Angeles within 15 meters of major roads. Would you like speed limits for these road segments?

Frequently Asked Questions

01 **How does Google Roads MCP handle gaps in GPS tracks?**

The `snap_to_roads` tool reconstructs the path by interpolating points between your input coordinates, creating a smoother, more continuous road geometry than raw data allows. This helps visualize the intended travel route.

02 **Can I use Google Roads MCP for individual point analysis?**

Yes. If you have scattered GPS points that don't form a clear path, `get_nearest_roads` treats each coordinate independently to find its closest road segment and associated place ID.

03 **What is the difference between snap_to_roads and get_snapped_speed_limits?**

`snap_to_roads` only returns the clean geometry and place IDs. `get_snapped_speed_limits`, however, performs both functions in a single call, giving you the speed limit data along with the mapped path.

04 **Does Google Roads MCP require continuous GPS data?**

No. It supports both continuous paths using `snap_to_roads` and discrete points using `get_nearest_roads`, making it versatile for various data collection scenarios.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"google-roads": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

Google Roads is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by Google Roads. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	Google Roads MCP
Server ID	`019d75a9-2756-70ad-b4d4-1926602cfa5f`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/google-roads.