Digital Eyes on the Road: Using Street Cameras to Verify Traffic Integrity and Detect Sybil Attacks at City Scale

Jhonatan Tavori, Mehmet Kerem Turkcan, Zoran Kostic, Salvatore Stolfo and Gil Zussman
Columbia University
New York, NY, USA
{j.tavori,mkt2126,zk2172}@columbia.edu, sal@cs.columbia.edu, gil.zussman@columbia.edu

Abstract

Crowdsourced navigation platforms infer congestion from GPS-derived telemetry, enabling real-time traffic estimation and control at city scale. However, because these systems rely on device-reported location traces, they are vulnerable to *Sybil attacks*, where an adversary injects fabricated "vehicles" and synthetic slowdowns using emulators, scripted clients, or device farms. Such attacks have been publicly demonstrated to create "ghost" cars and phantom congestion that distort the displayed traffic layer and influence routing decisions.

We propose using street cameras as a real-time validation layer for GPS-based traffic inference. Street cameras add an independent signal grounded in physical observation ("eyes on the road"), rather than device-reported telemetry alone. They also sit on a different attack surface than mobile devices: manipulating camera evidence targets physical infrastructure rather than the mobile client, and is often more expensive to scale than client-side Sybil injection. By cross-checking camera-observed vehicle activity against GPS-based mobile traces, camera feeds provide a conservative verification layer, and can be used to raise integrity alerts.

In this paper, we implement an end-to-end GPS-camera pipeline that (i) matches cameras to nearby road links, (ii) aligns and stabilizes the streams, (iii) scores mismatch using a residual-based detector to produce alerts, and (iv) extends coverage to blind spots using nearby intersections.

We evaluate the system using New York City DOT street cameras (200+ active in Manhattan) together with GPS-based traffic levels collected via the TomTom API. The pipeline achieves 91.5% detection under a moderate injected attack ($\delta = +0.05$) and reaches $\approx$99% detection for stronger attacks. We further address blind spots: using only neighbors, we retain a high-load consensus signal in 96.68% of Manhattan tested locations.

1 Introduction

Mobile navigation services estimate traffic from real-time telemetry reported by end-user devices (e.g., mobile phones) [15, 23]. Platforms aggregate location and speed samples, map-match them to road segments, and combine historical baselines to produce congestion overlays, ETAs, and routing recommendations. As these outputs shape navigation decisions for large numbers of users, and underpin driver-assistance systems and autonomous vehicles, the integrity of their inferred traffic state is a critical security and safety concern [4, 12].

(a) Actual state of the road.

(b) Reflected state (jammed) due to the attack.

Figure 1: Illustration of a Sybil-driven congestion attack. By injecting manipulated probe reports, the adversary forces the platform to account for congestion that is decoupled from the physical traffic conditions on the roadway.

Navigation services infer traffic from location and speed reports submitted by user devices, which creates an Sybil attack surface: an adversary can manipulate the traffic layer by controlling what devices report. The attacker can either (i) compromise devices (e.g., via malware or instrumentation) or (ii) avoid compromising anyone by generating many reporting identities using emulators, scripted clients, or device farms that replay or synthesize GPS traces. Prior work has demonstrated both stealthy GPS manipulation of a victim's reports [39], as well as Sybil attacks that inject coordinated 'ghost' vehicles to bias aggregate congestion inference on platforms such as *Waze* and *Google Maps* [3, 29, 33].

Figure 1 illustrates this threat: an attacker can inject 'ghost cars': reported vehicles that do not exist physically. The

'ghost cars' will make the platform infer congestion that is not present on the road. If the service believes a segment is jammed, it may reroute drivers away from it, shifting demand and reshaping traffic patterns beyond the targeted street. At city scale, this manipulation can impose large cumulative costs in wasted travel time, fuel, and operational disruption, and can undermine trust in routing infrastructure. These concerns become sharper as smart cities and autonomous vehicles increasingly rely on data-driven traffic layers for control, routing,and safety-critical planning.

Existing defenses largely focus on identifying or preventing *fake* traffic reports. In the Sybil setting, this typically means (i) hardening the reporting endpoint so an attacker cannot easily control what a device submits (e.g., platform/app attestation and anti-tampering), (ii) strengthening user or device authentication to raise the cost of creating many identities using emulators or device farms, and (iii) detecting Sybil behavior from structure in the reported telemetry, such as co-location graphs, trajectory consistency, or behavioral anomaly detection [12]. These mechanisms can raise attacker cost, but they still lack an external notion of truth: attestation does not prove physical location; authentication can still be bypassed at scale; and behavior-based filters can be evaded or confused by legitimate disruptions.

As report-based defenses can be bypassed, integrity checks should rely on evidence *grounded in the physical world*, not only on the same stream of device reports the attacker can shape. This motivates adding an independent sensing channel to corroborate the inferred traffic state.

In this paper, we demonstrate how to fuse street-camera sensing with GPS-derived traffic flow to improve the integrity of traffic state estimation. Figure 2 summarizes the end-to-end workflow: camera and GPS inputs are associated, aggregated and smoothed to reduce measurement noise, and then compared via residual-based scoring to produce an integrity signal (i.e., attack alerts).

We implement an end-to-end GPS-camera verification pipeline. The inputs are different modalities: camera-based vehicle counts and GPS-based traffic flow. We spatially match cameras to nearby road links and temporally align the asynchronous streams, which allows us to learn typical co-variation patterns. When the GPS-based congestion signal is artificially increased by an injected attack, the resulting cross-source mismatch grows and is detected by our residual-based monitor.

A key challenge is that the two sources measure different things: cameras yield vehicle-count time series, while the GPS feed reports a congestion-intensity value (analogous to the green/yellow/red traffic layer in navigation apps). Despite this mismatch, we associate each camera with nearby road links and align the asynchronous streams in time, allowing us to learn what "normal" agreement looks like at each location (using a residual-based monitor). When an injected attack

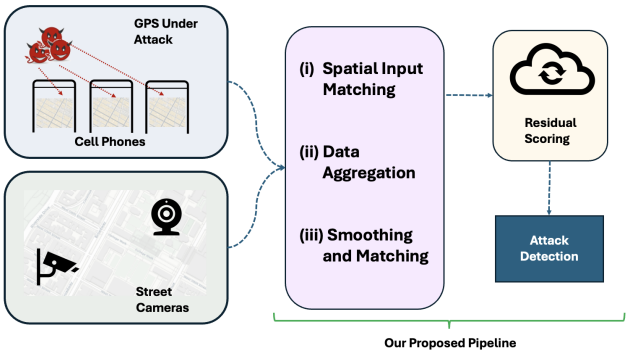


Figure 2: GPS and camera fusion workflow. We match each camera to nearby road links, align timestamps and signals if needed, and score mismatches to raise alerts.

inflates the GPS-reported congestion without a corresponding change in observed vehicle activity, the disagreement grows and is surfaced by our residual-based monitor.

To demonstrate feasibility at city scale, we use measurements from New York City and focus on the Manhattan borough, where dense camera coverage and complex road topology create a challenging urban setting. We use NYC Department of Transportation (DOT) traffic cameras [25], with over 200 active cameras in Manhattan providing vehicle-count time series at 15 s granularity. We pair these counts with GPS-derived traffic measurements from *TomTom* via the TomTom Traffic Flow API [31], which reports timestamped, link-level traffic indicators (`traffic_level`) over road geometries.

In the Manhattan setting, our pipeline achieves 91.5% detection under a moderate injected manipulation ($\delta = +0.05$) and reaches ≈99% detection for stronger injections. We also evaluate robustness to camera blind spots: in Manhattan 'leave one camera out' tests, neighboring intersections preserve a high-load consensus signal in 96.68% of time bins, providing a conservative integrity mechanism when a center camera is unavailable.

Paper structure. Section 2 summarizes how navigation platforms infer traffic and reviews Sybil and spoofing attacks, including public demonstrations. It further surveys prior defenses for location integrity, contributor authentication, and multi-source verification. Section 3 formalizes the threat model, mitigation goals, research questions, and practical challenges for camera-based integrity verification. Section 4 presents our cross-modal verification approach: data sources, camera processing, spatial association of cameras and road links, gap-aware temporal alignment, and the alerting logic used to surface sustained disagreement. Section 5 describes our implementation and experimental analysis. In particular, Section 5.1 details the GPS-camera verification pipeline (spatial matching, temporal alignment, and mismatch scoring) and demonstrates injected-manipulation detection, and Sec-

tion 5.2 evaluates robustness under partial camera coverage via a neighbor-consensus analysis, including case studies at two Manhattan intersections. Section 6 then evaluates both GPS-camera mismatch detection and blind-spot consensus at Manhattan scale, across over 250 cameras. Section 7 discusses limitations, deployment considerations, and privacy implications. Section 8 concludes with closing remarks and directions for future work.

2 Background and Related Work

In this section, we provide background on navigation platforms' reliance on GPS-based telemetry, Sybil attacks on traffic inference, and real-world demonstrations of these attacks. We then discuss related work on detecting and preventing Sybil manipulation in crowdsourced navigation systems.

2.1 Background: Navigation Platforms and Sybil Attacks

GPS telemetry in navigation platforms. GPS-derived telemetry is central to many smart-city applications and to emerging autonomous and connected-vehicle systems. Navigation platforms such as Google Maps and Waze estimate congestion by aggregating location and speed samples from large sets of devices [19]. They also combine recent observations with historical baselines and incidents. The result is a per-segment traffic state (often colored on map as green/yellow/red congestion) that impacts the estimation of ETAs and routing recommendations. Because this is driven by device-reported telemetry, an adversary who can inject, replay, or distort reports can bias the inferred traffic state and influence routing decisions.

Sybil injection and GPS manipulation. Navigation platforms can be manipulated either by injecting fabricated device data at the application layer or by distorting the location layer that produces those reports. In an application-layer Sybil attack, an adversary creates many identities (e.g., emulated devices, scripted clients, or coordinated "ghost" vehicles) that submit consistent location and speed traces so the service infers congestion on targeted segments [9]. Prior work has shown that such virtual attempts can cause 'phantom' congestion and trigger rerouting in systems such as Waze [36, 37], and similar manipulation opportunities have been analyzed for Google Maps' speed-based congestion inference [10]. GPS manipulation (including GNSS spoofing) targets the location layer directly by causing devices to report false positions and speeds. This then propagate through map-matching and can aggregate into the traffic layer [17, 30]. An attacker can pair Sybil-generated traces with GPS spoofing to make fabricated trajectories more realistic and harder to filter [39].

Real-world demonstrations of sybil attacks. These threats have been demonstrated outside controlled simulations. In March 2014, a student demonstration induced Waze to display a nonexistent traffic jam and to suggest alternate routes by creating many fake Android devices via emulation, generating coordinated slow trajectories, and submitting reports consistent with being stuck in traffic [29, 33]. The demonstration was conducted on campus roads to limit external impact, but it showed that large-scale aggregation of device reports can be influenced by an adversary who can scale credible identities. A subsequent academic study (MobiSys 2016) provided explicit before/after evidence: Waze initially recommended the target segment as the fastest route, then rerouted users once the injected congestion increased the segment's inferred travel time [36]. This linkage, from manipulated reports to routing decisions, is the central risk: false congestion can divert traffic, waste capacity, and create secondary congestion elsewhere.

In February 2020, the "99 phones" experiment demonstrated a different but operationally similar mechanism. Rather than emulation, an artist moved dozens of physical smartphones at walking speed, producing telemetry consistent with heavy slow traffic. After roughly an hour, Google Maps rendered an extended red congestion segment on an otherwise lightly trafficked road [3, 14]. Google publicly noted that traffic visualization is refreshed from aggregated anonymous data and did not (at that time) explicitly filter out that configuration [3]. Together, these demonstrations show that both virtual Sybil fleets and physical device multiplicity can bias GPS-based traffic inference, motivating defenses that corroborate inferred traffic state using independent, infrastructure-grounded sensing.

2.2 Related Work: Sybil Detection

Prior work has explored a range of defenses for improving the integrity of GPS-based and crowd-sourced traffic inference. These include mechanisms that make it harder to generate or sustain fake traffic reports (e.g., endpoint hardening and authentication), methods that detect Sybil behavior from reported patterns, and techniques that identify manipulation at the GPS/GNSS layer for such spoofing attacks. A recent survey of vehicular traffic security similarly organizes defenses into trust schemes, learning-based misbehavior detection, and authentication mechanisms, highlighting that most protection efforts focus on identifying or excluding malicious senders within the reporting channel [12]. These approaches make different assumptions and target different parts of the pipeline. We summarize the most relevant directions and use them to motivate our focus on street cameras as an additional, infrastructure-grounded source for validation.

A common direction is to filter *fake* traffic reports by raising the cost of producing many credible identities. Thus, server-side account controls, or platform operator, can reduce low-

effort attacks by requiring attackers to use real devices or more sophisticated bypasses [36, 37]. However, without trusted certification, an adversary can still present many identities and confuse the system [9], as attestation alone does not prove where a device actually is.

A complementary class of defenses tries to identify 'ghost' users by investigating the reported data. SybilRank [6]) apply graph-based techniques to separate Sybil clusters when they have limited connections to honest users. Waze-focused work proposed constructing such graphs from short-range encounters and using them for detection [36, 37]; [1] note these assumptions can break under real-device usage of the attacker and thus are not sufficient.

Other work detects manipulation from reported movement patterns. Trajectory- and behavior-based models look for what appears as implausible motion, coordination signatures, or inconsistent contribution patterns, including learning-based approaches applied to crowdsourced navigation [38]. These methods can catch naive attacks, but may be evaded by adaptive attackers. Further, legitimate disruptions (e.g., parades, storms, or closures) that naturally deviate from historical baselines, may also raise false alarms.

Regarding GNSS spoofing, which were mentioned as one of the attacking schemes, the literature mostly focuses on receiver- and signal-layer detection and cross-checking (e.g., anomaly detection at the GNSS signal level or redundancy across receivers) [18]. These techniques target location-layer manipulation, but they do not address application-layer Sybil fleets that fabricate traffic reports without relying on GNSS reception.

Limits of report-based defenses. Real-device farms can scale despite attestation, and rate limits can be averaged across many accounts and over time. Behavior-based detectors can be evaded by tuning traces to match expected distributions (e.g., plausible speeds, dispersion, and timing). A particularly challenging case is gradual manipulation that stays within the range of normal variability caused by rush hour dynamics, weather, or events, making it difficult to separate attacks from legitimate disruptions [1, 27].

Thus, in practice, a determined adversary can often bypass defenses that operate only on device-reported traffic data (perhaps at higher cost), since the attacker can shape the very signals those defenses rely on for detection.

These limitations motivate a complementary requirement: attacks alarms should be triggered by evidence that is independent of mobile device reports. When report-based controls are bypassed, the system needs a mechanism that is grounded in the physical world and harder to manipulate remotely at scale. Traffic-poisoning studies emphasize that effective countermeasures depend on an explicit attacker model and thresholds calibrated to realistic capabilities [27].

Multi-source verification and our contribution. To overcome the limitations of defenses that rely only on device-reported traffic data, we suggest cross-checking traffic estimates using independent sources so that manipulating one stream does not determine the inferred traffic state. In practice, this can mean suggest several implementations, including comparing GPS-based congestion with infrastructure telemetry or incident feeds. The value of such fusion depends on true independence: if multiple inputs ultimately derive from the same end-devices or report ecosystem, the manipulation can still propagate and bypass the comparison [18, 27].

In this work, we use street cameras as an infrastructure-grounded verification channel and compare camera-observed vehicle activity against GPS-derived traffic flow to flag sustained mismatches consistent with manipulation. In the following section, we describe the camera-based verification layer and the city-scale GPS-camera verification pipeline in detail.

Prior work on fusing GPS-derived probe signals with physical sensors has primarily focused on improving traffic estimation and operations [20]. In contrast, we use camera–GPS comparison in real time as an integrity mechanism: we explicitly monitor for disagreement to detect manipulation. We further demonstrate and evaluate this use at city scale across Manhattan.

3 Threat Model and System Goals

In this section, we describe the threat model considered by our pipeline, and state the operational goals of the proposed integrity-verification architecture. We then describe the main research questions that guide our design and evaluation.

Threat model. We consider an adversary whose objective is to artificially inflate *reported* congestion on targeted road segments. The attacker injects fake GPS telemetry at scale, for example, via emulator fleets, device farms, or scripted clients. By doing so, it poisons the platform's traffic inference and creates phantom congestion that is not present on the road.

The systemic risk extends beyond individual misroutes. Because algorithmic navigation influences large populations of drivers, and is increasingly incorporated into emerging autonomous-driving systems, poisoning the inferred traffic state can create cascading effects. Manipulation may cause visible disruption (e.g., widespread rerouting and localized jams), but it can also induce subtle, hard-to-detect inefficiencies; at city scale and over time, even small biases can accumulate into substantial societal costs in time and fuel.

This Sybil threat can be executed remotely and at relatively low cost, without physically affecting the roadway. Consequently, we focus on infrastructure-grounded "ground-truth" validation that remains effective even when application-layer GPS defenses are bypassed.

Mitigation goal and scope. Our goal is to build a city-scale integrity pipeline that can detect Sybil-style manipulation of mobile GPS reports by flagging disagreement between GPS-reported traffic conditions and camera-observed vehicle activity. Thus, we do not aim to replace primary Sybil defenses (e.g., attestation, identity friction, or behavior-based filtering). Instead, we provide an independent, infrastructure-grounded verification signal that supports conservative monitoring decisions when persistent cross-source inconsistencies arise.

Research questions. Our implementation is organized around three questions that together test whether the proposed verification pipeline is practical and effective at city scale.

- **(RQ1)** Can camera-derived *vehicle-count* signals, obtained from low-rate/resolution camera interfaces cities typically expose (i.e., periodic frames, not continuous video), be spatially matched and aligned with GPS-based traces well enough to enable cross-source validation?
- **(RQ2)** Can residual-based GPS-camera mismatch scoring detect targeted Sybil-style false congestion at city scale?
- **(RQ3)** If mismatch can be detected where cameras exist, can we extend verification to camera blind spots by using consensus from neighboring intersections when a center camera is missing or not installed?

4 Our Approach: Real-Time Pipeline for Traffic Verification with Street Cameras

In this section, we describe the inputs to our real-time verification pipeline and the practical challenges it must handle. Our design is motivated by the threat model in Section 3.

As reviewed in Section 2, most prior defenses operate within the reporting ecosystem (e.g., authentication or anomaly detection at the report-level). These mechanisms can raise the attacker's cost, but they still lack *physical* evidence of what is happening on the road. Since an adversary can manipulate GPS-derived reports at scale, integrity checks benefit from a validation signal that is not produced by the same reporting channel.

We therefore use street cameras as an infrastructure-based verification layer in a city-compatible setting: periodic camera frames paired with a commercial GPS-based traffic feed. Our pipeline cross-checks the two sources and raises an alert when disagreement persists beyond normal variability, producing a practical integrity signal for monitoring.

Pipeline Sources: Cameras and GPS Our goal is to build a verification pipeline that plugs into existing municipal sensing infrastructure. Because bandwidth constraints and privacy-oriented policies often limit what cities can expose, we restrict ourselves to the minimal public interface [24, 25]: periodic traffic-camera frames, rather than assuming continuous, high-rate streaming access. This design choice also defines a core technical challenge: the camera input yields vehicle *counts*, not direct speed estimates (as full video stream tracking might provide), so verification must relate count-based activity to GPS-derived congestion intensity.

The first input to our pipeline is, for each camera location, a timestamped vehicle-count time series paired with the camera's coordinates. To obtain these counts, we rely on the collection and processing pipeline of [11, 34], built over NYC Department of Transportation (DOT) public traffic cameras [25], and the COSMOS NSF testbed [11, 28]. The system of [34] ingests the DOT feeds and performs vehicle counting using YOLO-LR, an adaptation of YOLO optimized for low-resolution footage. This design is important for generality: NYC DOT streams are 352×240 pixels, well below standard object-detection dataset resolutions. Upstream processing in [34] was performed on a single compute node equipped with dual NVIDIA A100 40 GB GPUs, producing per-camera count series at 15 s granularity (with class-level counts) that we use as the cameras' input.

The second input is a GPS-derived traffic feed from the TomTom Developer Traffic Flow API [31], which provides real-time, link-level measurements aligned to road geometries. For each road segment, TomTom reports a timestamped traffic level value in the interval $[0, 1]$ that indicates the *relative* traffic level: it is calculated as the current speed on the segment expressed as a fraction of its free-flow speed, highlighting areas of congestion when the fraction drops. TomTom further classifies traffic conditions based on how much slower vehicles are moving compared to free-flow conditions [32]. We ingest and store the API stream per segment as provided, while preserving the original timestamps.

These two inputs are heterogeneous: cameras provide vehicle counts from fixed viewpoints, while TomTom provides a link-level congestion indicator derived from relative speed. Bridging this gap lies in the core design problem. In the remainder of this section, we describe (i) the sensing challenges of working with periodic frames and relating those counts to GPS-based congestion; and (ii) the spatial challenges of aligning cameras to the road map.

Sensing Challenge: Working From Frames and Matching Counts to GPS Because the camera interface is frame- or count-based, the main challenge is extracting a stable activity signal and making it comparable to the GPS congestion indicator through aggregation, smoothing, and careful missing-data handling.

Another sensing challenge is camera availability. DOT feeds are not uniformly continuous: outages and missing intervals vary across locations and days. In our pipeline, missing camera periods are treated as *no observation* (not as zero traffic), and we can exclude those timestamps from cross-source

scoring and, when needed, concatenate contiguous observation blocks rather than forcing a fully dense time series. This keeps the detector from being biased by gaps and allows the baseline relationship to be learned from the data that are actually observed. In addition, as we show in Section 5.2, neighboring intersections can often provide corroborating evidence when a given camera is unavailable, offering a conservative fallback under partial coverage.

At fine granularity, camera counts are noisy. We thus stabilize the camera signal before comparison by aggregating into coarser bins and applying rolling smoothing when needed.

Spatial Challenges: Camera Viewpoint and Area-Based Association A second challenge is spatial alignment. A street camera observes a particular viewing direction and a partial set of lanes, which often does not map cleanly to a single road segment, especially near intersections. Rather than assuming a one-to-one camera-road mapping, we treat each camera as representing a local *area* around its fixed coordinates and associate it with all nearby road links within a radius.

This area-based association is robust to viewpoint uncertainty and road geometry, and it provides a practical basis for comparing camera-observed vehicle activity with GPS-derived congestion on nearby links. Section 5 and Figure 3 detail the spatial association procedure.

In the next section, we show that these challenges can be handled at city scale: despite the mismatch between camera vehicle counts and GPS-derived congestion intensity, the fused signals support detection of injected Sybil-style manipulation. We also address camera blind spots by using neighboring intersections for consensus when direct camera observation is missing.

5 Implementation and Analysis

In this section we present the implementation and experimental analysis. In Section 5.1 we describe the GPS-camera verification pipeline and attack detection; in Section 5.2 we evaluate robustness to missing cameras via spatial neighbor consensus.

5.1 Cross-Modal (GPS-Camera) Fusion and Manipulation Detection

We next describe how we connect street cameras to GPS-derived traffic signals and use the resulting cross-check to detect an attack. Building on the challenges in Section 4, we suggest spatial association, stabilization, and residual-based mismatch scoring.

5.1.1 GPS-Camera Spatial Matching and Alignment

Our first step is to make the camera and GPS streams comparable in space and time. We associate each camera with nearby road links using TomTom road geometries: for each camera's latitude/longitude, we consider road polylines[1] in the traffic database and retain links whose geometry falls within the camera's local neighborhood.

As illustrated in Figure 3, we match a road link to camera i if a point along the link's polyline falls within a fixed radius of the camera's location. In our baseline configuration, we use a radius of 0.01 miles. This rule does not assume that the camera observes the entire link; Rather, it defines a conservative spatial neighborhood for comparison. This can be useful when the camera direction is unclear, just its location.

Given the matched link set for camera i, $\mathcal{L}(i)$, we compute a corresponding GPS signal by aggregating link-level values at each timestamp using the mean of the traffic level across the associated links:

$$\overline{\texttt{traffic\_level}}_i(t) = \frac{1}{|\mathcal{L}(i)|} \sum_{\ell \in \mathcal{L}(i)} \texttt{traffic\_level}_\ell(t).$$

In the Manhattan configuration, this procedure yields more than 50,000 unique associated links across 265 cameras.

After spatial aggregation, each camera i is paired with a corresponding GPS-derived signal summarizing traffic on its nearby road links. Yet the two sources are sampled very differently: camera counts arrive every 15 seconds, whereas TomTom actual values updates are asynchronous and may vary across links and time.

We therefore align the streams explicitly. We first transform the aggregated TomTom traffic level into a congestion-intensity signal,

$$g_i(t) = 1 - \overline{\texttt{traffic\_level}}_i(t),$$

so that larger values correspond to heavier congestion. We treat missing intervals as *no observation* (not as zero congestion) and align each camera count series with its corresponding $g_i(t)$ using nearest-neighbor (as-of) matching under a bounded time tolerance. Finally, to account for noise and differing update rates, we evaluate a range of uniform temporal aggregation and smoothing windows before computing cross-source agreement and mismatch scores.

5.1.2 Mismatch Scoring and Manipulation Detection

After the spatial matching and temporal alignment, we have for camera i, a paired observations $(c_i(t), g_i(t))$, where $c_i(t)$ is the camera-derived vehicle-activity signal and $g_i(t)$ is the corresponding GPS-based congestion-intensity signal aggregated from nearby matched road links at time t.

[1] A *polyline* is a piecewise linear curve which is defined by an ordered sequence of coordinates. Road networks, including TomTom's, use polylines to encode roads and routes [16].

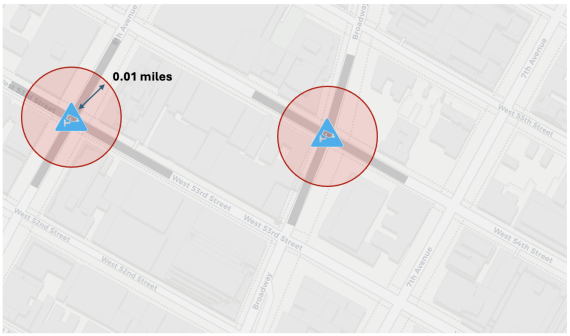


Figure 3: Camera-to-road (spatial) matching: camera (blue), spatial radius (red), associated road links (grey). A link is matched if its polyline point lies within 0.01 miles of the camera; the camera's GPS signal we treat is the aggregate `traffic_level` over matched links.

Because the two sources arrive at different cadences (camera counts at 15 seconds when available, TomTom may update asynchronously), we first aggregate the camera counts to time bins before comparison. We choose the smoothing window empirically: across Manhattan, longer smoothing windows produce more stable cross-source agreement, and we therefore use a long-minutes operating window in our evaluation.

For each camera, we fit a pre-attack baseline that captures the typical relationship between camera-observed vehicle counts and the corresponding GPS-derived traffic level. Let $z_i(t)$ denote the normalized residual at time t, i.e., the deviation between the observed GPS traffic level $g_i(t)$ and the counts observed by the cameras $c_i(t)$. Large positive $z_i(t)$ corresponds to where reported GPS congestion is higher than what the camera observations would support.

We then use a one-sided local (windowed) accumulation inspired by CUSUM/MOSUM control to detect mismatch between the two sources [13, 22, 26]. [2] We first filter the normalized residual signal $z_i(t)$ to retain only positive deviations above the noise floor (dictated by κ, the slack parameter):

$$u_i(t) = \max\big(0, z_i(t) - \kappa\big).$$

We then accumulate u_i over a rolling window (of length W):

$$S_i^{(W)}(t) = \sum_{\tau=t-W+1}^{t} u_i(\tau).$$

We use a finite window to limit long-memory effects. Because both inputs are noisy (even after binning and smoothing), the detector should emphasize recent persistent mismatch, and not to accumulate deviations that can cause delayed alarms based on history charts, nor to let short-lived visual glitches trigger alarms.

Because different cameras may experience different levels of natural variance, we normalize the accumulated result using a threshold h_i learned from pre-attack behavior, for each camera. The resulting alarm score is

$$A_i(t) = \frac{S_i^{(W)}(t)}{h_i}.$$

High $A_i(t)$ indicates persistent mismatch between the reported GPS congestion and camera-observed vehicle counts, that exceeds the camera's baseline. An alert is raised when $A_i(t)$ bypass a threshold (set to 1.2). Under our threat model, this pattern is consistent with Sybil-attack false congestion or related manipulation of GPS traffic reports, and we treat it as an integrity signal requiring inspection.

5.1.3 Evaluation Under Injected Sybil-Style Congestion

To model Sybil-driven false congestion, we inject an artificial increase into the GPS-derived congestion intensity after a fixed start time, while leaving the camera stream unchanged. This captures our target attack mode: the GPS-based feed is manipulated through fabricated reports, so it indicates elevated congestion even though camera-observed vehicle activity in the surrounding area does not change accordingly.

We use April 21–29 (2025) as the evaluation window, since this period provided consecutive camera coverage from the NYC DOT feeds. We activate the injected Sybil manipulation at April 26. To model the effect of false congestion on the GPS-derived signal, we add an offset δ to the traffic leveled provided by the GPS metrics (and bounded by the maximal value of 1):

$$g_i^{\text{attack}}(t) = \min\big(1,\, g_i(t) + \delta\big), \qquad t \geq t_0,$$

where t_0 is the attack start time and $\delta \in \{0.01, 0.025, 0.05, 0.10\}$ support several attack values, from moderate to severe. We illustrate the detector behavior on two representative Manhattan intersections, *8 Ave @ 14 St* and *1 Ave @ E Houston St*.

We compute two score trajectories under the same learned baseline and per-camera threshold: an *attacked* score (comparing the cameras to the manipulated GPS after the attack activation) and a *non-attacked* score (using the original GPS measures). Figure 4 reports results for two representative Manhattan intersections, *8 Ave @ 14 St* and *1 Ave @ E Houston St*. We use a decision threshold is of $(A_i(t) = 1.2)$. In both cases, the attack score (i.e., residual signals) increased after the attack activation, where larger attack magnitude leads to an earlier and stronger alarm triggering.

Overall, the results for both intersections are monotone in the attack magnitude, as post-activation attacked scores

[2] We use this detector because it accumulates only positive residual evidence (above a certain value), emphasizing the attack impact of upward shifts. The finite window avoids unbounded history effects and keeps pre/post behavior comparable [13, 22, 26].

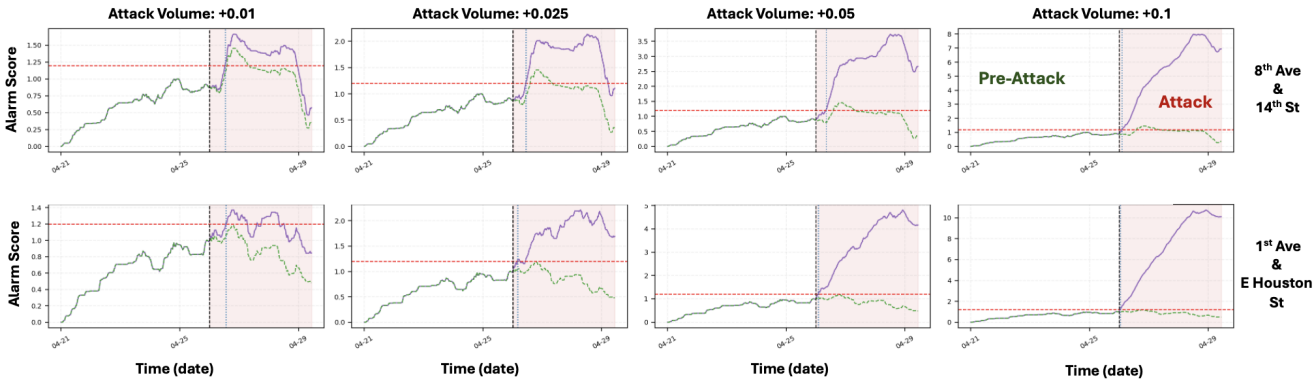


Figure 4: Pipeline detection results for injected Sybil attacks (Apr 21–29; attack activated Apr 26). Rows: *8 Ave @ 14 St* (top) and *1 Ave @ E Houston St* (bottom). Columns: attack volume $\delta = 0.01, 0.025, 0.05, 0.10$ (applied to GPS congestion and clipped at 1.0). Solid purple: attacked normalized score (relative residual signal), $A_i(t)$. Green dashed: non-attacked score under the same baseline and threshold. Red: alarm threshold (of 1.2). Larger values yield earlier and stronger alarm triggering.

increase when δ grows, while the non-attacked trajectories remain lower. This supports the pipeline's ability to distinguish manipulated regimes under injected Sybil GPS inflation.

5.2 Robustness to Missing Cameras via Spatial Neighbor Consensus

As discussed, road-camera coverage is incomplete and its availability can vary over time. This creates blind segments where GPS-reported traffic cannot be directly cross-checked against a physical camera. We consider an attacker who manages to manipulate GPS-reported congestion on such a 'blind segment'. Our solution is therefore a neighbor-consensus check: when the center camera is missing or unavailable, we use nearby intersections as corroborating evidence. If GPS indicates sustained congestion while surrounding cameras show no corresponding increase in vehicle activity, we flag the report as suspicious.

Neighbor consensus uses a common reliability principle in networked sensing: an isolated anomaly is treated as suspicious unless supported by nearby sensors within a spatiotemporal neighborhood. In wireless sensor networks, fault and outlier detection exploits local spatial correlation, using neighborhood agreement to separate real events from noise, faults, or adversarial injections [5, 40]. A similar concept appears in smart-grid security, where redundancy and cross-checking is carried to indicate a false-data injection [8, 21]. Transportation sensing applies similar logic for data-quality diagnostics and imputation: isolated anomalies are often treated as sensor faults unless corroborated by neighboring detectors [7, 35].

Our objective is slightly different: we use neighboring consensus as a mismatch integrity mechanism to validate GPS-reported congestion.

5.2.1 Neighbor Consensus Method Demonstration

We study two center locations: (i) *5 Ave @ 57 St* and (ii) *Amsterdam @ 72 St*. These cases illustrate two neighborhood structures. For *5 Ave @ 57 St*, we use nearby cameras along 57th Street and the adjacent avenues. For *Amsterdam @ 72 St*, we use a broader set of surrounding cameras spanning the local intersection neighborhood. Figure 5 depicts the intersections and selected neighbors.

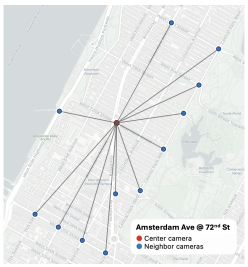
Note that we choose center locations that *do* have a working camera, so we can evaluate whether a neighbor-based consensus would have provide the same load magnitude which we can use to detect Sybil attacks. If neighbor agreement is strong in this setting, it supports using neighbors as a mismatch mechanism when the center feed is missing.

For each camera i, we smooth the vehicle-count stream and aggregate it into fixed-length bins. We then label a bin as high-load when its value is unusually high relative to that camera's baseline over the analysis window. Concretely, we apply this camera threshold scaled by a factor of $\alpha = 0.75$.

We then mark a center bin as *unique* if the center is considered high, but none of its neighboring cameras are considered high in the same bin or the immediately preceding bin (to allow for small timing offsets). Clearly, a low fraction of unique bins means that high-load events at the center are usually experienced by the nearby cameras, which supports using neighbors as an integrity signal when the center camera is unavailable.

Table 1 reports the number of center high traffic bins and the number of unique center bins. The observed unique rates are low. Thus, our pipeline will flag a blind-segment GPS report as suspicious if it indicates of high traffic level while neighboring cameras remain non-high around that timestamp.

(a) Case A: 5 Ave @ 57 St and surrounding cameras.



(b) Case B: Amsterdam @ 72 St and expanded surrounding set.

Figure 5: Two blind-spot case studies used for neighbor consensus analysis.

Intersection	# Nbrs	High Traffic	Unique	Rate
5 Ave @ 57 St	5	580	34	5.86%
Amsterdam @ 72 St	12	636	1	0.16%

Table 1: Neighbor consensus (5-min bins; 60-sample window; camera threshold scaled by a factor of $\alpha = 0.75$).

6 City-Scale Evaluation: Pipeline Demonstration across Manhattan

We next demonstrate that the two methods developed (in Sec. 5.1 and Sec. 5.2) scale beyond specific intersection examples: (i) GPS-camera mismatch can be used for Sybil attacks detection across hundreds of locations, and (ii) neighboring cameras across Manhattan can provide a conservative integrity signal when a camera is missing (blind spots). Within New York City, we focus on Manhattan as its camera density and road complexity provide a challenging urban setting.

6.1 Manhattan-Wide Attack Detection

We scale the GPS-camera mismatch detector and carry a Manhattan-wide run of the pipeline. We use the same pipeline and parameters as in Section 5.1: cameras are matched to nearby TomTom road links using a 0.01-mile radius, counts are resampled and smoothed, and mismatch is scored using the one-sided normalized windowed accumulation signal. We evaluate all Manhattan cameras with sufficient coverage before and after the attack activation during Apr 21–29.

A practical issue at this scale is that a small subset of locations is chronically unstable in this dataset (e.g., fragmented availability, blank intervals, or varied count behavior), which can trigger alarms even in the non-attacked baseline. Since such feeds are not usable for integrity monitoring without site-specific handling, we exclude such cameras. This yields a filtered evaluation set of 223 cameras out of 265 initially usable cameras (about 15% removed). We report the Manhattan-wide results on this filtered set.

We model a Sybil-style manipulation by inflating the GPS-derived congestion signal for each camera i out of the 223 relevant cameras. Concretely, we add an offset δ to the per-camera fused GPS congestion intensity $g_i(t)$ after the attack start time, creating an attacked metrics:

$$g_i^{\text{attack}}(t) = \min\big(1,\, g_i(t) + \delta\big), \qquad t \geq t_0,$$

where $t_0 =$ Apr 26, 2025 00:00. We range the attack volume $\delta \in \{0.01, 0.02, \ldots, 0.20\}$ to assess the impact of its volume on the detectability, and compute detection rates over the filtered camera set.

Figures 6 and 7 summarize the Manhattan-wide results. Figure 6 shows per-camera outcomes for a representative attack magnitude ($\delta = 0.05$), with 91.5% detection rate. Many of the remaining non-detections are spatially concentrated rather than uniformly distributed. For example, several cluster near the Queensboro Bridge corridor on the east side of Manhattan, which may reflect location-specific variability. Notably, as the injection magnitude increases, these areas also begin to trigger, and the non-detection region shrinks accordingly. Per Figure 7, which reports detection rates across the full range of the magnitudes, detection increases monotonically with attack magnitude: it rises from 18.8% at $\delta = 0.01$ and reaches $\approx 99\%$ by $\delta \approx 0.10$.

Overall, detection follows the same monotonic trend and saturates as the injected manipulation grows. These Manhattan-wide results show that GPS–camera mismatch monitoring remains effective when applied at city scale, across hundreds of intersections.

6.2 Manhattan-Wide Neighbor Consensus for Blind Spots

We next generalize the spatial neighbor consensus demonstration beyond the two intersections we studied in Section 5. We treat each camera i in Manhattan as a potentially missing center. We measure how often its high-load bins are *not* corroborated by surrounding cameras.

For each camera i, we take its five nearest neighboring cameras (by geographic distance) as the comparison set. We smooth each camera's vehicle-count signal (running average over samples) and aggregate it into bins. As before, a bin is marked high-load when the center camera's binned value is high relative to its own baseline.

We then mark a red center bin as *unique* if none of the neighbors are red in the same bin or in the immediately previous bin (to allow for small timing offsets). Finally, for each camera we summarize neighbor consensus using the fraction of high-traffic bins that are unique:

$$\text{UniqueRate}(c) = \frac{\#\text{ unique high-traffic bins at } i}{\#\text{ high-traffic bins at } i}.$$

Lower unique rates indicate *stronger* neighbor agreement.[3]

[3] Intuitively, a high-traffic bin is unique only when the center camera

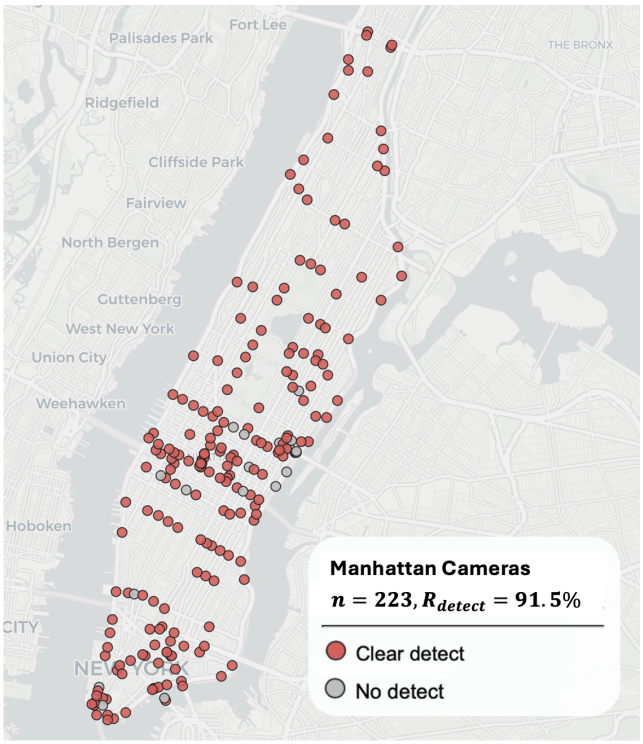


Figure 6: Per-camera outcomes at additive attack $\delta = 0.05$ on filtered Manhattan cameras ($n = 223$). Gray: no detection. red: (clear) detection.

Figures 8 and 9 summarize results for Apr 21–29, 2025 over the active Manhattan cameras. The city map in Fig. 8 shows that most locations are in the low-unique-rate regime, with a smaller set of outliers. Note that the remaining outliers are not uniformly distributed. Several appear near boundary regions such as bridge approaches and major tunnel entrances (e.g., the Midtown Tunnel), where the 'nearest neighbors' within Manhattan may not capture the same traffic dynamics. In these areas, the natural neighborhood for corroboration can span across borough boundaries.

The distribution/CDF view in Figure 9 confirms this quantitatively: the median unique rate is 3.32% (mean 8.67%), 73.62% of cameras have unique rate at most 10%, and 84.66% have unique rate at most 20%.

From a security perspective, these Manhattan-wide results support using nearby cameras as a mismatch signal when a center camera is missing. Locations with low unique rates can use stricter neighbor-based validation rules.

indicates high load but none of its neighbors do; thus, fewer unique bins means high-load periods are likely to be observed by at least one neighbor camera.

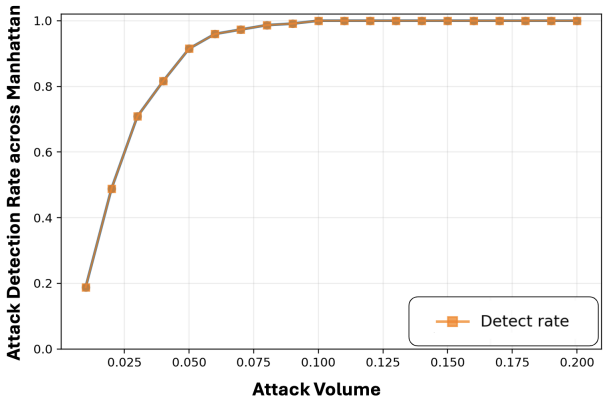


Figure 7: Manhattan-wide detection and clear-detection rates versus additive attack magnitude $\delta \in [0.01, 0.20]$ (filtered cameras, $n = 223$). Detection rates increase monotonically with attack magnitude and saturate for larger δ.

7 Discussions and Limitations

The Multi-Modal Approach to Integrity Validation Existing Sybil defenses (including endpoint attestation, behavioral filtering, and identity analysis; see Section 2) provide an important defense layer. The defense schemes can raise attacker cost and block low-effort manipulation. Yet, because these defenses still reason primarily over device-generated reports, they remain constrained by the fact that the attacker can control the reporting channel. A determined adversary is likely to still bypass these mechanisms using many real devices, or mimic benign mobility patterns closely enough to confuse detectors.

Our pipeline adds an independent verification layer using multi-modal approach. Street cameras provide infrastructure-grounded evidence from outside the GPS reporting channel: they observe actual vehicles at fixed locations and allow us to derive aggregate signals (vehicle counts) that can compare to GPS-derived congestion.

Combining the two data inputs increases attack cost directly. Now, to create false congestion without triggering an alert, an adversary must either manipulate device reports while also producing consistent physical effects, compromise infrastructure sensing, or exploit the cases where inputs are not available. While this does not guarantee perfect security, it provides a stronger mechanism than relying on device reports alone. Our Manhattan-wide evaluation further shows that this mechanism scales and remains useful even with natural camera coverage gaps.

Note that an attacker could also, in principle, target camera infrastructure or stage physical effects. Thus, this design on its own is not an absolute source of trust. However, this represents a different class of attack than Sybil injection, typically requiring localized presence and/or infrastructure intrusion rather than scalable remote data injection. In this paper, we

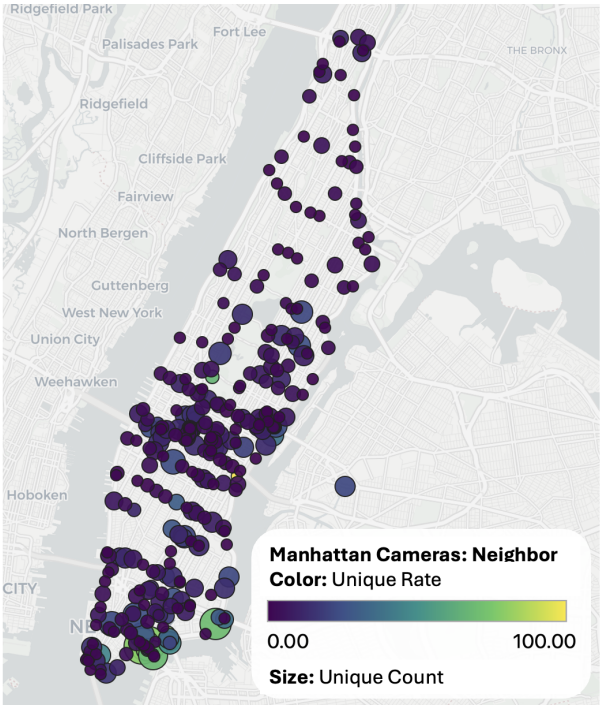


Figure 8: Manhattan-wide neighbor cross-validation ($k = 5$): per-camera unique-rate map. Color encodes unique rate, and marker size encodes unique-bin count. Most locations are in the low-unique regime, with a smaller set of spatial outliers.

do not attempt to defend against a strong adversary who can compromise municipal camera infrastructure, tamper with video feeds, or systematically stage physical traffic to match a fabricated congestion narrative. Those remain outside the scope of our threat model.

Privacy, Anonymity, and Data Handling. Camera-based sensing raises privacy concerns, particularly in municipal deployments where data may cross organizational boundaries, and especially where pedestrians are part of the environment. Our pipeline operates on aggregate vehicle counts; it requires neither storing identifiable imagery nor tracking vehicles across a sequence of cameras views. Nevertheless, real-world deployment requires clear data governance, including access controls, auditing, and transparency about what is collected and how it is used.

A practical requirement is that privacy protections must be enforced *before* any video or derived data leaves the local sensing domain. This aligns with prior work on the NSF PAWR COSMOS testbed, which demonstrated a real-time anonymization pipeline for smart-city intersections that removes sensitive identifiers (e.g., pedestrian faces and vehicle license plates) at the edge, enabling downstream analytics without exporting raw identifiable video [2, 11].

Accordingly, we recommend an 'edge-first' deployment

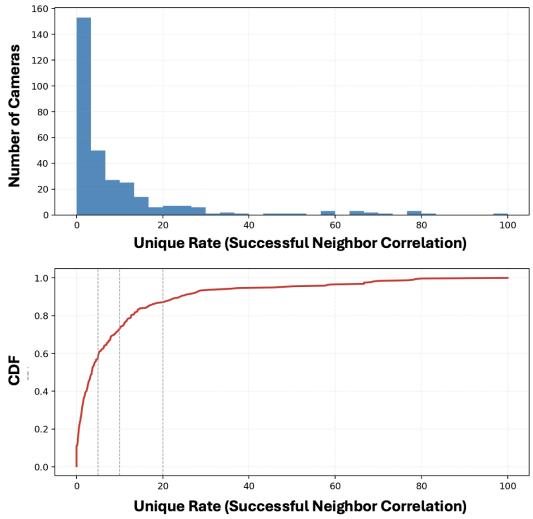


Figure 9: Manhattan-wide neighbor cross-validation ($k = 5$): distribution of unique rates (top) and cumulative distribution function (bottom). The mass near low values indicates that neighbor cross-validation is strong for most cameras, with a long but limited high-unique tail.

model for our pipeline. I.e., process raw camera feeds locally, and export only privacy-preserving aggregated data (in our case, per-camera vehicle-count time series) to the downstream mismatch detector.

Why not rely on cameras alone? A natural question is whether we could avoid integrity risks posed by mobile devices at all, by relying entirely on camera-based sensing. While street cameras provide strong, infrastructure-grounded evidence at specific locations, they do not replace GPS-derived telemetry. GPS feeds offer broader spatial coverage and different operational value: they support end-to-end routing and trip planning, and in many settings they provide demand-side context that cameras cannot directly observe (e.g., where vehicles intend to go, as captured within navigation apps). They also enable city-scale mobility statistics without depending on the availability of camera streams across jurisdictions.

For these reasons, our goal is not to replace GPS-based traffic inference, but to treat cameras as an additional verification source when integrity is in doubt.

8 Conclusions

In this paper, we addressed the integrity risk posed by Sybil attacks against GPS-based traffic inference in navigation platforms, including routing and planning for automated-driving systems. To add infrastructure-grounded evidence, we proposed using fixed street cameras as a real-time verification

layer for GPS-derived traffic signals.

We presented and implemented a city-scale GPS–camera pipeline that (i) associates each camera with nearby road links, (ii) aligns camera-derived vehicle counts with GPS-based congestion metrics, and (iii) scores cross-source mismatch using residual-based detection. We also addressed camera blind spots: since coverage can be incomplete or intermittent, we evaluated a neighbor-consensus test showing that surrounding intersections can often support conservative detection even when direct camera evidence at a target location is missing.

We evaluated the pipeline at city scale using New York City data (NYC DOT cameras and TomTom traffic levels). Across Manhattan, with more than 200 active cameras, the system achieves strong detection under injected GPS inflation: it reaches 91.5% clear detection at a moderate attack level and approaches 99% under stronger manipulations.

Overall, these results suggest that "digital eyes on the road" can complement report-based defenses by providing a deployable multi-modal detection signal for traffic inference. Future work includes evaluating additional attack strategies, extending coverage beyond dense urban cores, and integrating additional infrastructure signals to account for attacks on the cameras themselves.

Ethical Consideration This work does not raise any ethical concerns.

Acknowledgments

This research was supported in part by the NSF Center for Smart Streetscapes (CS3) under Grant EEC-2133516, NSF grant CNS-2450567, NSF grant CNS-2038984 and corresponding support from the Federal Highway Administration (FHWA), NSF grant CNS-2148128, funds from federal agency and industry partners as specified in the Resilient & Intelligent NextG Systems (RINGS) program, by the Fulbright Scholars Program, by the Fulbright Scholar Program, a program of the United States Department of State Bureau of Educational and Cultural Affairs and by the Zuckerman STEM Leadership Program.

References

[1] Lorenzo Alvisi, Allen Clement, Alessandro Epasto, Silvio Lattanzi, and Alessandro Panconesi. Sok: The evolution of sybil defense via social networks. In *Proc. of IEEE SP'13*, May 2013.

[2] Alex Angus, Zhuoxu Duan, Gil Zussman, and Zoran Kostić. Real-time video anonymization in smart city intersections. In *Prof. of IEEE MASS'22*, Oct. 2022.

[3] Brian Barrett. An artist used 99 phones to fake a Google maps traffic jam. Wired, February 2020. https://www.wired.com/story/99-phones-fake-google-maps-traffic-jam/.

[4] Neal E. Boudetter. Building a road map for the self-driving car. The New York Times, 2017.

[5] Joel W Branch, Chris Giannella, Boleslaw Szymanski, Ran Wolff, and Hillol Kargupta. In-network outlier detection in wireless sensor networks. *Knowledge and information systems*, 34(1):23–54, 2013.

[6] Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. Aiding the detection of fake accounts in large scale social online services. In *Proc. of NSDI'12*, Apr. 2012.

[7] Chao Chen, Jaimyoung Kwon, John Rice, Alexander Skabardonis, and Pravin Varaiya. Detecting errors and imputing missing data for single-loop surveillance systems. *Transportation Research Record*, 1855(1):160–167, 2003.

[8] Ruilong Deng, Gaoxi Xiao, Rongxing Lu, Hao Liang, and Athanasios V Vasilakos. False data injection on state estimation in power systems—attacks, impacts, and defense: A survey. *IEEE transactions on industrial informatics*, 13(2):411–423, 2016.

[9] John R Douceur. The sybil attack. In *International workshop on peer-to-peer systems*, pages 251–260. Springer, 2002.

[10] Cihan Eryonucu and Panos Papadimitratos. Sybil-based attacks on google maps or how to forge the image of city life. In *Proc. of ACM WiSec'22*, May 2022.

[11] Mahshid Ghasemi, Yongjie Fu, Xinyu Ouyang, Peiran Wang, Mehmet Kerem Turkcan, Jhonatan Tavori, Sofia Kleisarchaki, Thomas Calmant, Levent Gürgen, Zoran Kostic, et al. Real-time video analytics for urban safety: Deployment over edge and end devices. In *Proc. of ACM/IEEE SEC'25*, Dec. 2025.

[12] Amber Hankins, Tapadhir Das, Shamik Sengupta, and David Feil-Seifer. Eyes on the road: A survey on cyber attacks and defense solutions for vehicular ad-hoc networks. In *Proc. of IEEE CCWC'23 (Computing and Communication Workshop and Conference)*, Mar. 2023.

[13] Douglas M Hawkins and David H Olwell. *Cumulative sum charts and charting for quality improvement*. Springer Science & Business Media, 2012.

[14] Alex Hern. Berlin artist uses 99 phones to trick Google maps traffic jam alert. The Guardian, February 2020.

[15] Juan C Herrera, Daniel B Work, Ryan Herring, Xuegang Jeff Ban, Quinn Jacobson, and Alexandre M Bayen.

Evaluation of traffic data obtained via gps-enabled mobile phones: The mobile century field experiment. *Transportation Research Part C: Emerging Technologies*, 18(4):568–583, 2010.

[16] Albert S Huang, David Moore, Matthew Antone, Edwin Olson, and Seth Teller. Finding multiple lanes in urban road networks with vision and lidar. *Autonomous Robots*, 26(2):103–122, 2009.

[17] Todd E Humphreys, Brent M Ledvina, Mark L Psiaki, Brady W O'Hanlon, Paul M Kintner, et al. Assessing the spoofing threat: Development of a portable gps civilian spoofer. In *Proc. of ION GNSS'08 (International technical meeting of the satellite division of the ION)*, Sep. 2008.

[18] Kai Jansen, Matthias Schäfer, Daniel Moser, Vincent Lenders, Christina Pöpper, and Jens Schmitt. Crowd-gps-sec: Leveraging crowdsourcing to detect and localize gps spoofing attacks. In *Proc. of IEEE SP'18*, 2018.

[19] Tobias Jeske. Floating car data from smartphones: What google and waze know about you and how hackers can control traffic. *Proc. of the BlackHat Europe'13*, 2013.

[20] Kichun Jo, Keounyup Chu, and Myoungho Sunwoo. Interacting multiple model filter-based sensor fusion of gps with in-vehicle sensors for real-time vehicle positioning. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):329–343, 2011.

[21] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–33, 2011.

[22] Alexander Meier, Claudia Kirch, and Haeran Cho. mosum: A package for moving sums in change-point analysis. *Journal of Statistical Software*, 97:1–42, 2021.

[23] Sashank Narain, Aanjhan Ranganathan, and Guevara Noubir. Security of gps/ins based on-road location tracking systems. In *Proc. of IEEE SP'19*, May 2019.

[24] NYC Department of Transportation. New engalnd 511 cameras. http://www.newengland511.org/cctv/.

[25] NYC Department of Transportation (DOT). Real-time traffic cameras (rtti). https://nyctmc.org/.

[26] Ewan S Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

[27] Simone Raponi, Savio Sciancalepore, Gabriele Oligeri, and Roberto Di Pietro. Road traffic poisoning of navigation apps: Threats and countermeasures. *IEEE Security & Privacy*, 20(3):71–79, 2021.

[28] Dipankar Raychaudhuri, Ivan Seskar, Gil Zussman, Thanasis Korakis, Dan Kilper, Tingjun Chen, Jakub Kolodziejski, Michael Sherman, Zoran Kostic, Xiaoxiong Gu, et al. Challenge: Cosmos: A city-scale programmable testbed for experimentation with advanced wireless. In *Proc. of ACM MOBICOM'20*, Sep. 2020.

[29] Meital Ben Sinai, Nimrod Partush, Shir Yadid, and Eran Yahav. Exploiting social navigation. In *BlackHat Asia 2015*.

[30] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. On the requirements for successful gps spoofing attacks. In *Proc. of ACM CCS'11*, Oct. 2011.

[31] TomTom. Tomtom traffic flow api. https://developer.tomtom.com/.

[32] TomTom. Traffic flow segment definition. https://developer.tomtom.com/traffic-api/documentation/tomtom-maps/traffic-flow/flow-segment-data.

[33] Nicholas Tufnell. Students hack Waze, send in army of traffic bots. Wired, March 2014. https://www.wired.com/story/waze-hacked-fake-traffic-jam/.

[34] Mehmet Kerem Turkcan, Jhonatan Tavori, Javad Ghaderi, Gil Zussman, Zoran Kostic, and Andrew Smyth. A vision-based analysis of congestion pricing in new york city. *arXiv preprint arXiv:2602.03015*, 2026.

[35] Lelitha Vanajakshi and LR Rilett. Loop detector data diagnostics based on conservation-of-vehicles principle. *Transportation research record*, 1870(1):162–169, 2004.

[36] Gang Wang, Bolun Wang, Tianyi Wang, Ana Nika, Haitao Zheng, and Ben Y Zhao. Defending against sybil devices in crowdsourced mapping services. In *Proc. of ACM MobiSys'16*, Jun. 2016.

[37] Gang Wang, Bolun Wang, Tianyi Wang, Ana Nika, Haitao Zheng, and Ben Y Zhao. Ghost riders: Sybil attacks on crowdsourced mobile mapping services. *IEEE/ACM transactions on networking*, 26(3), 2018.

[38] James J. Q. Yu. Sybil attack identification for crowdsourced navigation: A self-supervised deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4622–4634, 2021.

[39] Kexiong Curtis Zeng, Shinan Liu, Yuanchao Shu, Dong Wang, Haoyu Li, Yanzhi Dou, Gang Wang, and Yaling Yang. All your {GPS} are belong to us: Towards stealthy manipulation of road navigation systems. In *Proc. of USENIX Security'18*, Aug. 2018.

[40] Yang Zhang, Nirvana Meratnia, and Paul Havinga. Outlier detection techniques for wireless sensor networks: A survey. *IEEE communications surveys & tutorials*, 12(2):159–170, 2010.