Improving GNSS Positioning Correction Using Proximal Policy Optimization

Scott Nonweiler
Aerospace Engineering Sciences
University of Colorado Boulder
Scott.Nonweiler@colorado.edu

Joohan Chun
Aerospace Engineering Sciences
University of Colorado Boulder
joch9258@colorado.edu

Abstract— **Smartphone GNSS positioning, especially using Single Point Positioning (SPP) based solely on GPS L1 pseudoranges, often suffers from errors in the tens of meters due to multipath, signal blockages, and limited antenna quality. This paper proposes a data-driven correction technique using Proximal Policy Optimization (PPO), a reinforcement learning algorithm, to post-process SPP results and reduce positioning errors. The correction problem was framed as a Partially Observable Markov Decision Process (POMDP), enabling a policy to learn optimal corrections without relying on physical models. Using data from the Google Smartphone Decimeter Challenge (GSDC 2023), our approach demonstrates proof-of-concept accuracy by directly learning from past positioning errors and applying corrective actions in a continuous 2D spatial domain.**

Keywords— GNSS, smartphone positioning, reinforcement learning, POMDP, PPO, RTKLIB, GSDC

I. Introduction

Smartphones have revolutionized access to Global Navigation Satellite System (GNSS) positioning, yet their accuracy remains limited due to low-quality antennas, multipath, signal blockages, and noise-dominated hardware. In particular, Single Point Positioning (SPP), which relies solely on GPS L1 pseudorange measurements, often suffers from errors exceeding tens of meters. This poses a critical challenge for smartphone-based navigation and location-dependent services in urban and dynamic environments.

To address this problem, we propose a data-driven approach to correct SPP results using a reinforcement learning framework. Specifically, we model the GNSS correction problem as a Partially Observable Markov Decision Process (POMDP). The motivation for this formulation arises from the real-world nature of GNSS positioning: although the receiver obtains position estimates (observations), the true location of the user (state) is unobservable, and the measurement process is inherently stochastic due to atmospheric delays, hardware errors, and environmental factors.

However, the GNSS correction problem diverges in key ways from standard POMDP formulations. First, the state space — representing the user's position — is continuous and unbounded, unlike the discrete state spaces used in classical POMDP examples such as the tiger or cancer POMDP problems. Second, the action space, defined as the correction vector applied to the SPP output, does not influence future observations through a dynamic transition model; each timestep is independently observed and corrected based on a precomputed SPP sequence. This means the correction action at time *t* does not affect the state at *t+1*, rendering the traditional Bellman recursion inapplicable.

To overcome this limitation, a traditional error penalty reward function was modified by introducing a sequential penalty term that discourages erratic corrections across time and a direction penalty term that discourages large changes in direction from the initial SPP estimates. Specifically, we define the reward as a combination of negative penalties for instantaneous positioning error (between corrected estimate and ground truth), the temporal difference between consecutive corrections, and the divergence in heading from consecutive SPP and consecutive correction estimates. The temporal difference term encourages the policy to maintain smooth corrections while minimizing position errors, and the direction difference term ensures that the corrected estimates track the turns in the SPP estimate path accurately.

The algorithm selected to solve this problem formulation was Proximal Policy Optimization (PPO). Proximal Policy Optimization utilizes a policy network to output parameters of a probability distribution over continuous actions, then subsequently perform policy gradient ascent on the distribution. While PPO is traditionally used for online planning methods, the SPP problem formulation is inherently a formulation in which the actions, and consequently the policy, do not affect the state evolution over time. This makes the implementation of PPO akin to behavioral cloning with a shaped reward.

In this project, we use the RTKLIB software to compute SPP solutions from raw GNSS measurement and leverage the GSDC 2023 dataset, which provides GNSS logs and high-accuracy ground truth trajectories for training and evaluation [1, 2]. We define three levels of success for solving the correction problem:

- **Level 1** — Learn a policy that correlates positioning accuracy by correcting SPP outputs with the GSDC 2023 training set which includes ground truth.
- **Level 2** — Apply the trained policy to the test dataset which lacks ground truth. No reward is

computed; only the learned policy is used to perform correction.

- **Level 3** — Treat RTKLIB PPK (Post-Processed Kinematic) result as pseudo-ground-truth and evaluate the POMDP-corrected SPP accuracy compared to PPK.

By formulating GNSS correction as a modified POMDP and applying reinforcement learning to this real-world stochastic problem, we aim to improve smartphone GNSS accuracy using a scalable, data-driven method that can adapt to varying environments and devices.

II. Background and Related Work

Traditional methods to improve GNSS positioning include sensor fusion (e.g., using IMUs), statistical filtering (e.g., Kalman filters), and error modeling. However, these approaches typically require explicit models of the system or access to additional sensor data. In contrast, recent advances in data-driven and learning-based GNSS correction, particularly for low-cost devices, leverage supervised learning or imitation learning.

Our work builds on this direction by leveraging reinforcement learning (RL) in a POMDP framework to model the stochastic and partially observable nature of the GNSS correction problem. Unlike previous methods that rely on static mappings or handcrafted features, PPO enables adaptive learning of corrections with policy optimization directly guided by performance-based rewards.

To train and evaluate this method, GSDC 2023 dataset was utilized, which provides raw GNSS measurements from Android smartphones alongside high-accuracy ground truth data. Since the GSDC data consists of unprocessed raw measurements and smartphones do not output broadcast ephemeris directly, IGS broadcast ephemeris files were retrieved to enable proper positioning.

The raw data was first converted to the RINEX format using RTKLIB utilities. This step is critical to enable standardized processing of GNSS data across multiple receivers and satellite constellations. After conversion, SPP was performed using RTKLIB’s least squares engine based solely on GPS L1 pseudorange measurements. The output is a sequence of SPP positions that serve as the observation to our reinforcement learning correction system.

The pseudorange model used in SPP consists of the true range between satellite and receiver, as well as various error terms including satellite and receiver clock bias, ionospheric delay, tropospheric delay, and multipath. Smartphones exacerbate these errors due to limited antenna quality and high noise levels. As a result, raw SPP solutions can suffer from tens of meters of error, especially in urban environments. This motivates the need for a robust correction mechanism such as the one proposed in this paper.

III. Problem Formulation

We consider the GNSS correction task as a decision-making problem under partial observability. The core idea is to train a policy that learns output correction vectors that adjust SPP outputs toward ground-truth positions.

We formulate the GNSS correction task as a Partially Observable Markov Decision Process (POMDP), motivated by the fact that the true position of the receiver is not directly observable, and the observations (i.e., SPP outputs) are corrupted by various sources of GNSS error. However, this problem diverges in key aspects from classical POMDP settings.

In a traditional POMDP, the agent observes noisy outputs and takes actions that influence future states through a known or unknown transition model. In our case, although the agent observes noisy SPP positions, the sequence of future positions is predetermined and unaffected by any action the agent takes. Each timestep is treated independently, and the action—defined as a correction vector to be applied to the SPP estimate—has no impact on future observations. This unique property invalidates the use of methods like DQN or POMCP, which depend on a dynamic environment and the evolution of state under a policy. Our correction policy must instead operate in a fixed, non-interactive environment.

Despite this, the problem remains partially observable because the SPP-derived observation is not the true state. The ground truth is available only in training data and is used to compute the reward during learning. During inference (e.g., test time), the agent must act based solely on the current noisy observation.

Although the real-world receiver state exists in three dimensions (x, y, z), this paper restricts the analysis to 2D (x, y) for tractability. This simplification still captures most of the positioning error variance found in smartphone GNSS data, particularly in urban environments where lateral positioning is more critical.

We define the elements of the POMDP as follows:

- ***State space***: The true but unobservable receiver position at time t.
- ***Observation***: The 2D SPP output (x, y) at time t.
- ***Action space***: A continuous 2D correction vector (Δx, Δy) applied to the observation.
- ***Rewards***: The negative Euclidean distance between the corrected observation and the known ground truth, the negative temporal difference between corrected position estimates, and the negative weighted angle heading difference above a five degree threshold
- ***Transition***: Nondeterministic and unavailable. The transition model is determined solely by the movement of the GPS receiver over time and is not affected the POMDP agent’s actions

Because of these properties, our POMDP is effectively decomposed into a series of static correction problems rather than a temporally coupled sequence. This required us to depart from value iteration-based methods and instead adopt policy gradient reinforcement learning, specifically Proximal Policy Optimization (PPO), which can learn a policy in this simplified yet nontrivial setting [3, 4, 5].

IV. Solution Approach

To address the inherent noise and non-linearity in smartphone-based GNSS SPP, we adopt Proximal Policy Optimization (PPO)—a stable and widely used on-policy reinforcement learning algorithm. PPO was chosen over other RL methods (e.g., DQN, POMCP) because our correction environment lacks state transitions: each GNSS observation is temporally independent, and the agent cannot influence future states. This structure invalidates traditional value iteration or transition-based decision models, making PPO's direct policy optimization suitable.

Our correction framework treats each observation as a 2D vector in Earth-Centered Earth-Fixed (ECEF) coordinates (converted from latitude and longitude), and the action space as a continuous 2D vector representing the correction in meters. The agent outputs a Δx, Δy vector which is applied to the SPP position to reduce positioning error.

The model was trained on an input vector containing 8 variables. The variables include the SPP x estimate in ECEF coordinates, the SPP y estimate in ECEF coordinates, the standard deviations for x and y SPP estimates, and four SPP temporal difference variables. Two of the SPP temporal difference variables are the difference between the current SPP estimate and the previous (time t-1) SPP estimate in x and y values, and the other two SPP temporal difference variables are the x and y difference between the time t-1 SPP estimate and the time t-2 SPP estimate.

We used an open-source PPO implementation in PyTorch[3], modified to suit continuous action environments. The ActorCritic network comprises fully connected layers with Tanh activations and separate heads for actor (policy) and critic (value function). The policy outputs mean vectors for a multivariate Gaussian distribution, with a learnable action standard deviation that decays over time.

The GNSSCorrectionEnv environment was custom-designed using OpenAI Gym interface, incorporating real SPP outputs and ground truth positions from the GSDC 2023 dataset. Each episode corresponds to a trajectory from one drive, with each step involving correction of one GNSS fix.

The observation space O includes:

- x, y : ECEF coordinates from SPP output
- σ_x, σ_y: Standard deviation estimates (converted to ECEF from ENU)
- $dx = x^t - x^{t-1}, dy = y^t - y^{t-1}$
- $dx_2 = x^{t-1} - x^{t-2}, dy_2 = y^{t-1} - y^{t-2}$

The action A is a correction vector $\Delta x, \Delta y$ bounded in range $[-10, 10]$ meters.

The reward function R was defined as:

$$-|x_{corrected} - x_{true}| - \lambda_1 |x^t_{corrected} - x^{t-1}_{corrected}| - \lambda_2 cos\theta$$

where λ_1 and λ_2 are the weights for the consistency and direction penalties, respectively, and $cos\theta$ is the cosine of the angle between the ground truth displacement vector ($x^t_{true} - x^{t-1}_{true}$) and the corrected displacement vector ($x^t_{corrected} - x^{t-1}_{corrected}$), computed as their dot product divided by the product of their magnitudes.

The reward function is composed of three terms: the first term penalizes the Euclidean error between the corrected and ground-truth positions, the second term penalizes sudden changes across consecutive corrections (temporal smoothness), and the third term penalizes deviations in direction headings between the SPP and corrected solutions. Initially, only the Euclidean error and temporal consistency were considered. However, this led to two main issues: first, the agent often failed to correct properly when the trajectory involved directional changes; and second, even in straight-line movement, it frequently adjusted the position in the opposite direction of the actual error, as the reward was based solely on distance. To address these shortcomings, we introduced the directional consistency term to guide corrections in the correct heading. The temporal smoothness term ($\lambda = 0.1$) prevents overfitting to individual noisy SPP errors and encourages consistent policy behavior. The weight for the directional terms was also set to 0.1. The following hyperparameters were used in training the GNSS correction model:

- Learning rate: Actor = 0.0001, Critic = 0.0003
- Action standard deviation: initialized at 1.0, decayed by 0.01 every 10,000 timesteps, down to a minimum of 0.05
- Training schedule: each trajectory was trained for 300 episodes
- Continuous action space (2D correction vector)

V. Results

We first trained the PPO algorithm using training data that included ground truth positions. After sufficient training, we evaluated the performance of the trained model on a test dataset that did not include ground truth data.

Figure 1 presents the results of applying the PPO algorithm on the training dataset. The trajectories of the Ground Truth, SPP, and PPO outputs nearly overlap, indicating that the PPO model learned to closely replicate the SPP trajectory and remained consistent with the ground truth. This visual alignment suggests that the PPO model converged well on the training data.

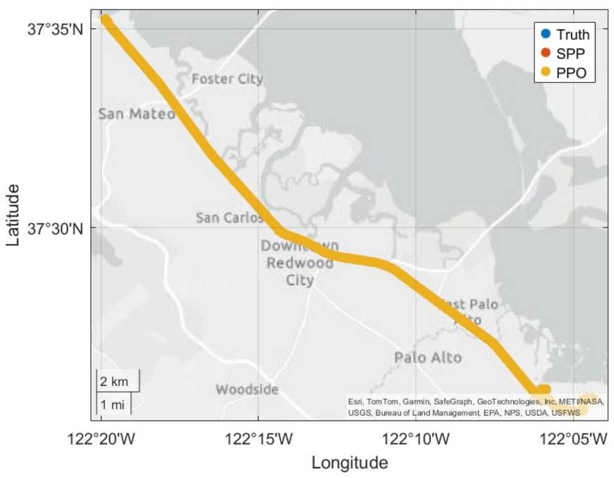
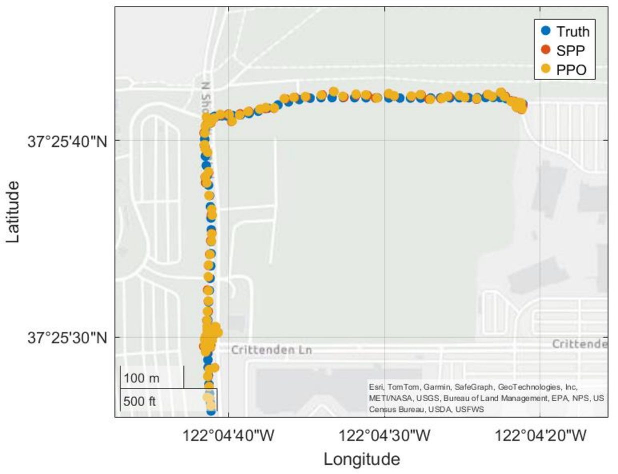


Fig. 1. Training result



Truth
SPP
PPO
37°25'40"N
37°25'30"N
Latitude
122°04'40"W
122°04'30"W
122°04'20"W
Longitude
Crittenden Ln
100 m
500 ft
Esri, TomTom, Garmin, SafeGraph, GeoTechnologies, Inc, METI/NASA, USGS, Bureau of Land Management, EPA, NPS, US Census Bureau, USDA, USFWS

Fig. 2. Zoomed in training result

Figure 2 shows a magnified view of the trajectory in Figure 1. It demonstrates that the inclusion of a direction penalty in the reward function enabled the model to apply reasonable corrections even during abrupt changes in direction. In earlier experiments where the direction penalty was not considered, the model failed to make appropriate corrections in such turning segments, resulting in accumulated error and degraded positioning performance.

A quantitative comparison of positioning accuracy is provided in Figure 3, which shows the empirical cumulative distribution function (CDF) of the horizontal error norm for both SPP and PPO with respect to the ground truth. The 50th percentile horizontal error for SPP was 21.29 meters, while PPO recorded a slightly higher error of 23.86 meters. Both methods showed an identical 95th percentile error of 34.79 meters. These results indicate that although the PPO model performed similarly to SPP on average, it exhibited a slight bias and did not outperform the baseline.

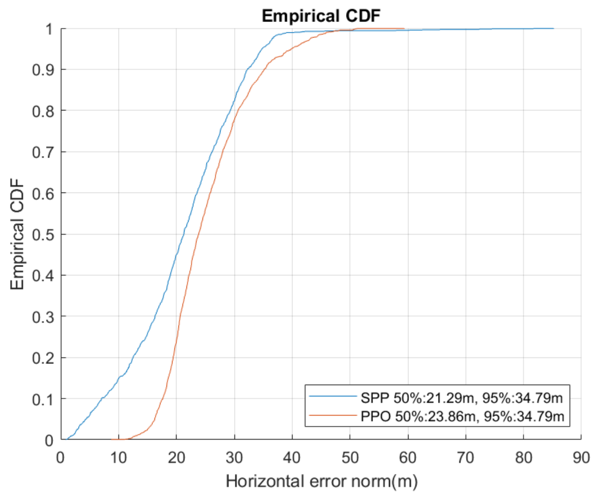
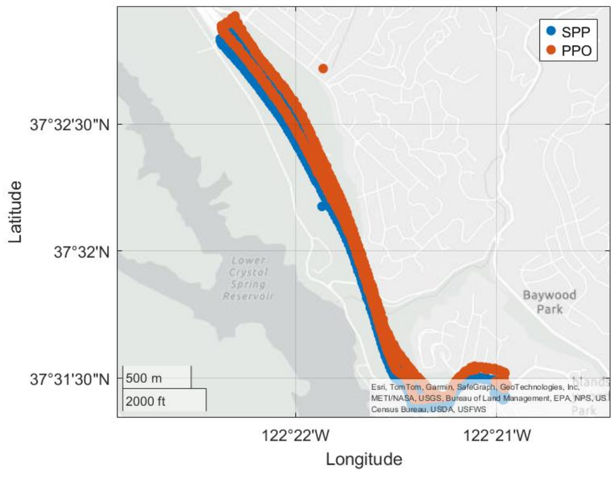


Fig. 3. Empirical CDF comparison

To evaluate the generalization capability of the trained PPO model, we applied it to a novel test set from the GSDC without providing any ground truth or rewards during inference. The result, shown in Figures 4 and 5, demonstrates that while the PPO-predicted trajectory followed the overall shape of the SPP path, a consistent positional bias was observed. Of note, however, is that the application of the PPO model to the novel training data set did result in smoothing of SPP estimates in locations where the SPP estimate covariance was high. Because SPP position estimates at these GPS-degraded locations vary widely but tend to be centered about the ground truth, this would have resulted in a more accurate PPO position approximation of the ground truth if the constant error had not been present. We note that the smoothing behavior was markedly more pronounced in the test case than when the PPO model was run on the training data, which accounts for the less accurate PPO estimates observed in Figure 1.



SPP
PPO
37°32'30"N
37°32'N
37°31'30"N
Latitude
122°22'W
122°21'W
Longitude
Lower Crystal Spring Reservoir
Baywood Park
500 m
2000 ft
Esri, TomTom, Garmin, SafeGraph, GeoTechnologies, Inc, METI/NASA, USGS, Bureau of Land Management, EPA, NPS, US Census Bureau, USDA, USFWS

Fig. 4. Test result

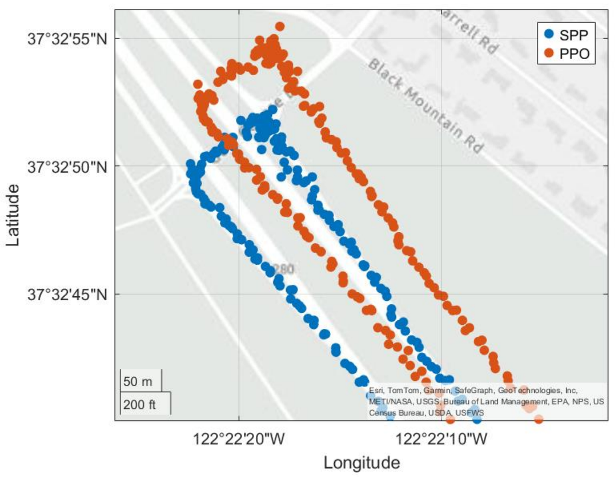

37°32'55"N
37°32'50"N
37°32'45"N
Latitude
122°22'20"W
122°22'10"W
Longitude
● SPP
● PPO
Black Mountain Rd
Farrell Rd
50 m
200 ft
Esri, TomTom, Garmin, SafeGraph, GeoTechnologies, Inc, METI/NASA, USGS, Bureau of Land Management, EPA, NPS, US Census Bureau, USDA, USFWS

Fig. 5. Zoomed in test result

We believe that there are four main factors contributing to the limited performance observed in our PPO-based correction approach.

First, although we initially intended to train the model across multiple trajectories, we ultimately chose to train on a single trajectory for 300 episodes. This decision was made because the PPO reward resets with each new trajectory, which hindered cumulative learning. We expect that if rewards could be accumulated across multiple trajectories, or across a single extended trajectory with a complex path, the model could achieve better generalization and improved performance.

Second, the inherent limitations of PPO in offline settings likely contributed to the poor results. PPO is an on-policy algorithm that assumes the training data is collected from the current policy. However, GNSS correction is fundamentally an offline problem where all observations are pre-collected. This mismatch can lead to unstable policy updates and hinder convergence.

Third, the performance could have potentially improved had we identified more optimal weight values for the temporal difference and directional penalty terms in the reward function. Improper weighting may have either under-constrained or overly penalized certain corrections, limiting the model's effectiveness.

Finally, our current implementation relies on SPP outputs and estimated standard deviations computed via RTKLIB as the observation space. We believe that if the system had instead been implemented fully in a Python or Julia environment—allowing the use of additional raw measurements such as Doppler or C/N0, and combining them with a Kalman filter-based position estimator—more informative observations and better corrections could have been achieved.

VI. Conclusion

Applying PPO to the selected problem proved to be more difficult than anticipated, and the SPP solutions for the provided data from Google were fairly accurate on average, meaning that improvements on the provided estimates required a nontrivial degree of optimization in the training of the PPO model. When plotting the ground truth, SPP solution, and PPO solutions together, it was observed that the PPO solutions did not differ significantly from the SPP solutions in areas with high fidelity SPP solutions. This is evidenced by the low average difference of 4.34m between the SPP solutions and the PPO solutions and is a satisfaction of our Level 1 benchmark for success in the project. This training was done without the ground truth being included as a parameter passed as part of the input vector, which represents a realistic POMDP scenario.

One area in which we had trouble and invested significant time was training the model to correlate to both the shape and the position of the training data. In our PPO implementation, the model would fit to the shape of the data, but output a translated / scaled, and ultimately inaccurate, solution. This ultimately prompted us to augment the input vector of the PPO model to include two previous SPP estimates and to include the direction penalty term in the reward function. The direction penalty term facilitated the model successfully outputting an accurate SPP modification as its own estimate, but also resulted in difficulties in encouraging the model to deviate from the SPP estimates where the SPP estimates were inconsistent or had high deviation in the estimate between adjacent points. Our goal was that when SPP estimates are inaccurate and significantly deviate from a straight or slightly curving path. We could utilize PPO to correct back to the ground truth that was available before and after the GPS-degraded section. It effectively interpolates the errant SPP estimates back to the ground truth track without the tradeoffs that a smoothing algorithm would introduce on a set of GPS solutions for a given path (for example, turning 90 degree turns into smoother arcs). The introduction of the direction penalty term allowed the model to track the SPP estimates with high fidelity, but it also discouraged the model from deviating from the SPP estimates, even in areas where the standard deviation of SPP was high, relatively speaking. One possible modification that we were not able to explore would be the effect of scaling the penalty factor for direction divergence based on the standard deviation of the SPP estimates provided in the input vector. If the penalty factor for direction divergence was inversely proportional to the SPP standard deviations, the model would have more leeway to diverge from SPP estimates in degraded environments, and the temporal difference penalty term would potentially incentivize the model to smooth the data in that section.

In attempting to generalize the PPO model from the training data to test data by applying the model to beforehand unseen data, the PPO model was off by a constant factor but was able to match the overall shape of the novel data set. Additionally, the PPO model exhibited the desired smoothing of the data when the SPP measurements were inaccurate that we desired. This generalization to the test cases showed promise, and we believe that if we were able to train the model on more extensive data sets or find an input parameter that provided effective "anchoring" of the estimate that we would be able to generate PPO estimates that demonstrably improve on SPP estimates. One potential option for helping localize the PPO estimate on the SPP estimate would be passing the first SPP estimate as a constant input parameter in the input vector.

VII. Contributions and Release

The authors grant permission for this work to be posted publicly. Joohan processed the raw GNSS measurements from the 2023 GSDC dataset into SPP solutions and implemented the PPO training and wrote the introduction and methods of the paper. Scott implemented Behavioral Cloning training using PyTorch, which ultimately was not included in the report because his computer broke over the weekend and he could not refine the code and generate graphs, and he and Joohan used Joohan's computer to refine the PPO code. Scott wrote the majority of the results and conclusions section of the report. Both Joohan and Scott helped iteratively improve the PPO algorithm after implementation, provided input and help with code implementation to the other person, and contributed to writing the remainder of the report.

References

[1] Rtklibexplorer, "Rtklibexplorer/RTKLIB: A version of RTKLIB optimized for low cost GNSS receivers, especially u-blox receivers. it is based on RTKLIB 2.4.3 and is kept reasonably closely synced to that branch. this software is provided 'as is' without any warranties of any kind so please be careful, especially if using it in any kind of real-time application.," GitHub, https://github.com/rtklibexplorer/RTKLIB (accessed May 5, 2025).

[2] "Google smartphone decimeter challenge 2023-2024," Kaggle, https://www.kaggle.com/competitions/smartphone-decimeter-2023 (accessed May 5, 2025).

[3] Comparative analysis of A3C and PPO algorithms in reinforcement learning: A survey on general environments | IEEE Journals & Magazine | IEEE Xplore, https://ieeexplore.ieee.org/document/10703056/ (accessed May 5, 2025).

[4] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv.org, https://arxiv.org/abs/1707.06347 (accessed May 5, 2025).

[5] J. Tang et al., "Improving GNSS positioning correction using deep reinforcement learning with an adaptive reward augmentation method," NAVIGATION: Journal of the Institute of Navigation, https://navi.ion.org/content/71/4/navi.667 (accessed May 5, 2025).

[6] Barhate, N. (n.d.). Nikhilbarhate99/PPO-PyTorch: "Minimal implementation of clipped objective proximal policy optimization (PPO) in Pytorch." GitHub. https://github.com/nikhilbarhate99/PPO-PyTorch

[7] Increasing GPS localization accuracy with Reinforcement Learning | IEEE Journals & Magazine | IEEE Xplore, https://ieeexplore.ieee.org/document/8998202/ (accessed May 5, 2025).