Accurate Real-Time Relative Localization Using Single-Frequency GPS

Will Hedgecock, Miklos Maroti, Akos Ledeczi, Peter Volgyesi, and Rueben Banalagay
Institute for Software Integrated Systems, Vanderbilt University
Nashville, TN, USA
{will.hedgecock, miklos.maroti, akos.ledeczi, peter.volgyesi, rbanalagay}@vanderbilt.edu

Abstract

For outdoor navigation, GPS provides the most widely-used means of node localization; however, the level of accuracy provided by low-cost receivers is typically insufficient for use in high-precision applications. Additionally, many of these applications do not require precise *absolute* Earth coordinates, but rather rely on *relative* positioning to infer information about the geometric configuration of the constituent nodes in a system. This paper presents a novel approach that uses GPS to derive relative location information for a scalable network of single-frequency receivers. Networked nodes share their raw satellite observations, enabling each node to localize its neighbors in a pairwise fashion as opposed to computing its own standalone position. Random and systematic errors are mitigated in novel ways, challenging long-standing beliefs that precision GPS systems require extensive stationary calibration times or complex equipment configurations. In addition to presenting the mathematical basis for our technique, a working prototype is developed, enabling experimental evaluation of several real-world test scenarios. The results of these experiments indicate sub-meter relative positioning accuracy under various conditions and in varying environments. This represents up to order of magnitude increase in precision over existing absolute positioning techniques or other unimodal GPS-based solutions.

Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-Based Systems**]: Real-Time and Embedded Systems—*GPS*

General Terms

Algorithms, Theory

Keywords

GPS, Relative Localization, Wireless Sensor Networks

SenSys'14, November 3–5, 2014, Memphis, TN, USA.

http://dx.doi.org/10.1145/2668332.2668379

1 Introduction

A wide variety of today's sensing applications comprise networks of mobile devices that rely heavily on highly accurate node location information. For outdoor applications, GPS is regarded as the clear choice for location determination, especially as its price and power requirements steadily decline; however, a significant tradeoff exists between cost and accuracy. Low-cost receivers, such as those found in smartphones and other handheld devices, can exhibit large errors on the order of tens of meters, especially in challenging RF environments. Higher quality devices provide better accuracy at the expense of price, with commercial-grade surveying equipment costing thousands of dollars while requiring extensive setup and calibration before becoming usable.

GPS enjoys widespread applicability in the positioning domain due to its unique ability to simultaneously service an infinite number of users and provide absolute coordinates anywhere on Earth. In many applications, however, the precise *relative* locations of the nodes are much more important than the precision of their absolute coordinates. For example, a common class of Wireless Sensor Network (WSN) applications rely on the distributed sensing of some signal (acoustic, RF, magnetic, etc.) to estimate its source location. It is easy to show that small errors in the sensors' relative positions can magnify in the sensor fusion, e.g., using multilateration when the source is outside the sensor network, even when the mean absolute position error of the sensors is zero. On the other hand, if the network geometry is perfectly known, but the entire network is shifted in absolute terms, the positioning error will show up in the estimated source location without any magnification. Furthermore, in many applications the end result is requested relative to the WSN in the first place, for example, in the case of smartphones or wearable sensors. There also exist numerous applications beyond WSNs that require accurate relative locations, including autonomous driving, collision avoidance, land surveying, precision agriculture, and formation flying of unmanned aerial vehicles (UAVs), among others.

The research in this paper aims to bridge the gap between cost and accuracy in *outdoor* localization by presenting a novel relative localization technique based on the Global Positioning System and only low-cost, single-frequency GPS receivers. The goal is to enable commercial receivers to achieve an unprecedented decimeter-scale level of accuracy for applications that require node locations in a relative coordinate

frame. Such an increase in relative accuracy without a corresponding increase in cost should enable the production of novel applications that are either uneconomical today or have proven to be out of reach given the current state of the art. It must be noted that the use of GPS is still too costly and power hungry for the cheapest and most resource constrained WSN platforms, e.g., the traditional motes. Our target is the mid- to high-end sensor and other wireless platforms including smartphones.

In our approach, we shift focus from the typical standalone GPS paradigm by allowing a network of receivers to share raw satellite measurements such that each participating node is able to localize all other nodes in the network using pairwise combinations of satellite observations. As such, each node is able to create an internal mapping of the locations of each of the "remote" nodes in terms of a set of 3D position vectors with respect to its own local coordinate system. This symmetrical approach to relative localization ensures that no single points of failure are present in the system and further allows for the addition or removal of sensor nodes on the fly with no effect on system usability.

The methods used to carry out these high-precision localization calculations comprise the **A**FV (Ambiguity Function Value) **P**eak **T**racking technique, or **APT** for short, which is introduced in this paper. It is well known that many GPS error sources are correlated over similar geographic regions; as such, simple differential techniques *should* provide an increase in accuracy over single-receiver algorithms. The corresponding increase in accuracy, however, is insufficient to reach the desired decimeter-scale level of precision due to uncorrected dual-receiver error sources, most notably caused by poorly synchronized receiver clocks. We explore methods of mitigating these additional error sources, with the end result that our novel localization technique is able to achieve relative positioning accuracies up to an order of magnitude better than existing methods that rely only on the correlations between absolute positions.

In a benign environment with clear views of the sky, our technique was able to achieve anywhere between a 7-33x improvement over standard GPS positioning algorithms, with relative precision on the scale of centimeters. Additionally, our method produced results ranging from a 2-3x improvement in moderately obstructed and highly dynamic environments, with precision on the order of decimeters, regardless of the baseline length between a pair of receivers, which, in our experiments, varied from 0 m all the way up to 3.5 km.

The rest of this paper is organized as follows. We start with a brief overview of the previous and related works leading to the research in this paper, including dual-receiver relativity errors, methods of dealing with such errors, a new temporal observation model, and a multi-receiver relative tracking algorithm. We then continue with an in-depth theoretical discussion of our novel localization technique. This is followed by a short description of the hardware prototype and software implementation used to carry out our research. We continue with an evaluation of the system and an analysis of experimental results and conclude by discussing the limitations of the approach and describing ideas for future continuation of the work.

2 Related Work

While standalone GPS positioning techniques have been refined and dramatically improved over the past few decades, research has begun to shift focus toward increasing the accuracy of GPS-based coordinates through use of either *multi-modal* sensing techniques or *multi-receiver* error mitigation techniques. In terms of multi-modal localization, research has been focused on the integration of GPS with an Inertial Navigation System (INS) [22] to provide *continuous* positioning data to fill in the gaps between the *discrete* GPS measurement intervals. Oftentimes, input from these multiple sensors are fused using a Kalman filter-based approach to produce a statistically optimal estimate of the underlying system state given a time series of noisy sensor measurements [19]. Research has also been applied to improving statistical position estimation by taking into account non-linearities in the GPS system model itself via use of "Unscented" or Sigma-Point Kalman Filtering techniques [21]. The most relevant research related to our work, however, involves use of data from *multiple* GPS sensors combined in novel ways to mitigate errors inherent to the GPS signals. Generally, these types of techniques fall into one of two classifications: Differential GPS (DGPS) or Real-Time Kinematic (RTK) navigation.

In most cases, an algorithm is considered to be DGPS-based if it produces a baseline vector between two receivers where one "reference node" remains static at a known location and a second node roams freely around the reference [16, 3]. The reference is erected in a pre-surveyed location with a clear view of the sky. Once it begins receiving satellite ranging signals, it retransmits them to nearby receivers. Any number of roving nodes can be dispatched to receive their own GPS ranging signals along with the ranging signals from the reference. In such a way, every node has two full sets of ranging data and a known reference position at every epoch. Using this information in a relative context, many of the large sources of error in the satellite ranging signals can be minimized or eliminated to produce a much more accurate baseline. Also, since the location of the reference node is precisely known, these baselines translate directly into absolute coordinates which makes DGPS an extremely attractive method for surveying.

There are, however, several caveats to using this type of methodology. The first and most obvious is the necessity of having a reference beacon with a precisely known location. It requires some amount of pre-planning and set-up to find an acceptable location for such a beacon, as well as to allow the beacon to localize its own position with high enough accuracy that it can be used as a reference for the roving nodes. This makes it a bad choice for on-the-fly type applications which cannot tolerate long set-up and calibration times [3]. The second caveat is that data from multiple epochs are usually required to achieve the desired level of precision with these techniques. As such, the roving receivers must remain stationary at each measurement point for a length of time in order to collect the appropriate amount of data. Due to the stop-and-go nature of these types of methods, they are more suitable for applications that use GPS localization primarily as a surveying tool, such as feature mapping or boundary determination.

Another methodology that shares many similarities with DGPS is the Satellite-Based Augmentation System (SBAS)[4]. The SBAS used with GPS is called the Wide-Area Augmentation System (WAAS), and like DGPS, it attempts to increase accuracy by estimating the various sources of error in the satellite observables. One major difference between DGPS and WAAS is that WAAS gives actual estimates for the contributions of individual error sources instead of lumping them together. This is necessary because the system serves a much wider area than DGPS, and the assumption of correlated errors under DGPS does not necessarily hold true for WAAS. WAAS determines its corrections from a large amount of data recorded by various monitoring stations throughout the US. However, since these corrections are based on very sparsely located stations, the level of accuracy achievable by this system is nowhere near as precise as a local DGPS setup.

A recent extension to standard DGPS methods that aligns more closely with the techniques presented in this paper is a classification known as Real-Time Kinematic (RTK) Positioning [15, 17]. This type of positioning represents a group of algorithms that uses a highly accurate (but ambiguous) GPS observable known as the *carrier phase* from two or more receivers to provide real-time localization updates. While RTK can be viewed as a subset of DGPS, it is usually considered separately, with the main difference being that RTK algorithms are **necessarily** carrier-phase based and always provide instantaneous and continuous receiver coordinates. The primary observable in DGPS is an unambiguous, but far less accurate measurement known as the *pseudorange*. Additionally, DGPS is concerned with minimizing correlated errors such that a more accurate *absolute* position can be estimated, whereas RTK is concerned with dealing with the ambiguities in the carrier phase measurements to provide more accurate *relative* localization results [12, 5, 3].

In the majority of cases, RTK algorithms will require an initial, stationary calibration phase in which to converge to a highly accurate estimate of the baselines between a reference node and some number of mobile receivers. The roving receivers are then free to move anywhere they like, and assuming that consistent locks on at least three satellites are able to be retained throughout the localization process, the accuracy is unlikely to degrade [3, 5]. It should be noted that the downside to DGPS and RTK algorithms alike is that they both presuppose the presence of a stationary reference node. The case where the locations of **both** nodes is unknown or, even worse, when the positions are unknown and both receivers may be moving, represents a realm of localization that has not yet been satisfactorily solved. The best approach in these cases at the time of writing is a so-called *moving baseline* method in which two receivers are assumed to be in motion, and an RTK-like algorithm attempts to estimate the baseline vector between them at every time epoch [11, 10, 8]. Most current algorithms which fall under this heading are proprietary in nature, and there is yet to be seen an open, standard method for dealing with the uncertainties in this problem. The technique offered by this paper aims to fill this gap in relative localization techniques.

3 Previous Work

In a previous work (see [7]), we developed a novel relative tracking algorithm known as RegTrack (**Re**lative **G**PS **Track**ing) which enabled centimeter-scale pairwise *tracking* of sets of mobile GPS receivers over a significant period of time. The research presented in this paper is a direct continuation of that work; as such, we summarize the three main concepts that were introduced in [7], including:

1. Identification and characterization of error sources specific to the *dual-receiver* localization case,
2. Extrapolation of satellite measurement data from independent GPS receivers to a common time epoch, and
3. Creation of a temporal observation model describing the relative changes in range between a satellite and two receivers through time.

Due to the imprecision of local clocks used in low-cost GPS receivers, it was found that the *actual* measurement times of observation sets taken at the same *apparent* time by two independent receivers may differ by up to several milliseconds. The high orbital velocities of the GPS satellites, coupled with the rotation of both the Earth and the local coordinate frames of the GPS receivers themselves during these seemingly small timing discrepancies, caused non-trivial relativistic errors to be introduced into the standard GPS positioning models when combined in a relative positioning sense.

In order to overcome these errors, we first characterized them as either *signal propagation errors* or *time bias errors* and quantified their respective effects on relative positioning accuracy. We then developed a correction procedure which could be used to extrapolate the satellite orbital positions, raw measurement data, and receiver coordinate frames into a cohesive unit, with the end result that the relativistic effects that originally introduced errors on the order of 1-3 meters per satellite observation now result in only micrometer-scale inaccuracies.

Using the corrected satellite observations, we developed a so-called *temporal double-differencing* model which is able to very precisely track the relative changes in range between a single satellite and two receivers through time. By noting that these changes in satellite-receiver range could be geometrically linked to the relative change in baseline between two receivers, we developed an algorithm whereby sets of receiver-receiver baselines could be accurately tracked without requiring knowledge of the precise absolute locations of any of the receivers taking part in the localization procedure.

While the accuracies achievable by our RegTrack system showed significant improvement over existing GPS-only solutions, our tracking approach suffered from one serious limitation, namely that it required **manual** baseline initialization. Additionally, if a complete loss of satellite locks was experienced at any point during the tracking procedure, manual re-initialization would again have to be carried out each and every time. As such, we set out to extend our RegTrack algorithm to carry out its own initialization routines, such that it could be used as a complete, standalone localization service. The results of that research are the topic of the remainder of this paper.

4 Theoretical Approach

The tracking results from the RegTrack algorithm described in the previous section provide exceptional accuracy when viewed from a dead-reckoning point of view; however, their real utility comes from allowing data from multiple epochs of time to be used in a single localization algorithm regardless of any motion between the receivers.

In order to discuss the theory in this section, a few concepts must be introduced. Firstly, the name of the observable used in our localization approach is the "carrier phase," which can be viewed as a high-precision estimate of the range between a satellite and receiver in terms of the number of wavelengths, or carrier cycles, of the GPS signal itself. The caveat to using this type of observation is that it is an *ambiguous* measurement. It is only able to report the cumulative changes in satellite-receiver range after the satellite lock has been established. The initial range between the satellite and receiver is an unknown constant which must be estimated along with the coordinates of the relative baseline vector that we are trying to determine. This can be exactly likened to the problem we faced with the results of our RegTrack algorithm; namely, we are presented with extremely precise navigation results, but we do not know the initial baseline between our two receivers when the tracking algorithm began. As such, the location of the track with respect to a reference node is ambiguous, including some unknown but constant bias equal to the original baseline between the two receivers at the start of the tracking procedure.

The standard model for carrier phase therefore includes a constant ambiguity term for each satellite which must be resolved if we are to use the observation directly (for example, in a trilateration procedure). A common technique to remove the majority of quantifiable error from a satellite observation is to perform an operation known as "double-differencing," whereby the observations from two different receivers, a and b, to one satellite, s, are subtracted from one another, and this result is subtracted from the same operation carried out on a reference satellite, s_0. In addition to causing all satellite- and receiver-specific errors to cancel out, the fractional portion of the constant ambiguity value also cancels out, guaranteeing that it is now an integer. The resulting double-differenced model looks like so:

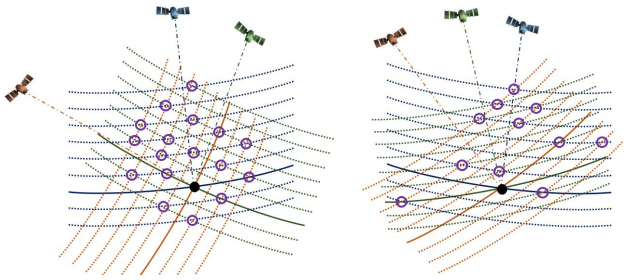
$$\nabla\Delta R_{a,b}^{s_0,s} = \lambda(\nabla\Delta\phi_{a,b}^{s_0,s} + \nabla\Delta N_{a,b}^{s_0,s}) - \nabla\Delta\varepsilon_{a,b}^{s_0,s} \qquad (1)$$

where $\nabla\Delta$ denotes the double-differencing operation, R is the satellite-receiver range, ϕ is the carrier phase observation, N is the integer ambiguity, λ is the wavelength of the carrier signal ($\sim$19 cm), and ε is any remaining error, dominated primarily in this case by multipath. Note that this equation only holds for as long as a constant satellite lock is maintained. There are occasions when the lock may be lost for such a small amount of time that the receiver is unable to detect it. In this case, a phenomenon known as a "cycle slip" occurs whereby the carrier phase observable increases or decreases by some number of whole carrier cycles. This affects Equation 1 above by adding or subtracting the corresponding number of slipped cycles to the integer ambiguity term. Since these slips are difficult to detect, they have the ability to wreak havoc on algorithms operating under the assumption that the integer ambiguity has remained constant through time.

Now that we have introduced the carrier phase concepts used in our research, we form the basis for our relative localization algorithm from the following observations:

1. Accurate tracking results allow us to use localization techniques that would normally be reserved for stationary network topologies.
2. The standard double-differencing observation model provides the strongest geometric solution while including the minimum amount of unmodeled error possible; however, it requires resolution of the so-called *integer ambiguities* (N term in Equation 1).
3. A solution relying solely on the resolution of ambiguities in the carrier phase model is inadvisable, since it will, by necessity, be susceptible to large errors due to undetected cycle slips and will mandate that a certain subset of satellites remain consistently visible through time.
4. The correct relative baseline position will be characterized by a set of ambiguity values that results in low error residuals when the *modeled* receiver-satellite ranges are calculated and compared to the *actual* satellite observations over a significant period of time.

This last point is best understood through an illustration:



(a) Set of potential receiver locations at time t_0.

(b) Set of potential receiver locations at time t_n.

Figure 1: Change in candidate node locations through time

We can think of the carrier phase observation to each satellite as a reported range value plus an unknown bias which is **guaranteed** to be equal to some number of carrier cycle wavelengths ($\sim$19 cm), where the actual receiver position must lie on one of the grid lines representing the reported range plus the ambiguity bias. From Figure 1a, it is apparent that there are multiple intersecting points which could all be potential receiver positions (the purple circles) since they are located on one of the gridlines from each of the satellites, and the relevant gridlines intersect one another very nearly perfectly. However, as the satellites move over time, the slopes of the gridlines will necessarily change such that a new set of possible locations becomes apparent (Figure 1b).

We can see that one and only one of the candidate locations remains constant (the black dot) through the change in satellite geometry. This persistent candidate location obviously corresponds to the correct receiver position. As stated

in our list of observations above, this can be verified mathematically by noting that the satellite-receiver ranges calculated using the incorrect ambiguity values (lines) from Figure 1a will no longer intersect with one another when recalculated after a change in satellite geometry. In other words, only the correct receiver position (with the correct ambiguity values) will have a continually low error residual when the computed satellite-receiver ranges are compared to the satellite observation values through time.

While this illustration shows the strength of the double-differenced observation model, we have already said that it can suffer severe drawbacks in that:

- Receivers must remain stationary while the correct integer ambiguities are being resolved, as random motion introduces 3 additional unknowns per unit time in a 3D coordinate space.
- This stationary calibration phase is typically quite lengthy because the GPS satellites are such a long distance from the Earth that the requisite change in satellite geometry occurs relatively slowly over time.
- Cycle slips are difficult to detect and will result in either very high error residuals or very wrong position estimates.
- The double-differenced model includes use of a reference satellite that may go in and out of visibility, undermining the ability of the ambiguity values to be resolved since their values are completely dependent on the reference satellite being used.

As such, we decided to investigate methods of carrying out our localization procedure without requiring resolution of the integer ambiguities.

4.1 Ambiguity Function Method Overview

From literature, we found a mathematical concept introduced in 1981 by Counselman and Gourevitch called the "Ambiguity Function Method" (AFM) [1]. This method is unique in that it leverages the "integer-ness" of the ambiguity values in the double-differenced carrier phase model to determine a baseline position which minimizes the range errors in the participating satellites' observations. Recall the double-differenced carrier phase model from Equation 1. By isolating the integer ambiguity term on the left-hand side of the equation (and neglecting the error term for the moment), we get:

$$\nabla\Delta N_{a,b}^{s_0,s} = \frac{\nabla\Delta R_{a,b}^{s_0,s}}{\lambda} - \nabla\Delta\phi_{a,b}^{s_0,s} \tag{2}$$

This shows that at the *correct* baseline coordinates, the entire right-hand side of the above equation will be a perfect integer (in the absence of any errors). Mathematically, we can test a value for its integer-ness by converting the number to radians and taking its cosine:

$$\cos(2\pi \cdot \text{integer}) \equiv 1.0 \tag{3}$$

For numbers that are not perfect integers, the resulting value will drift further and further from 1.0, reaching a minimum of -1.0 at any integer $\pm$ 0.5, or in other words, at the least "integer-like" value possible. Extending this concept to the double-differenced carrier phase model, we see that:

$$\cos(2\pi \cdot \nabla\Delta N_{a,b}^{s_0,s}) = 1.0 \tag{4}$$

And by extension, at the correct 3D baseline coordinates:

$$\cos(2\pi \cdot [\frac{\nabla\Delta R_{a,b}^{s_0,s}}{\lambda} - \nabla\Delta\phi_{a,b}^{s_0,s}]) = 1.0 \tag{5}$$

This is an interesting concept in that it allows us to search for a baseline solution in the *position domain*, instead of in the *measurement* or *ambiguity* domains, as is usually the case.

By summing the resulting cosine values for all possible satellite observations and dividing this result by the total number of observations, n, we can calculate an *Ambiguity Function Value* (AFV) in the range from $-1.0 \leq \text{AFV} \leq 1.0$:

$$\text{AFV} = \frac{1}{n}\sum_{i=1}^{n}\cos(2\pi \cdot [\frac{\nabla\Delta R_{a,b}^{s_0,i}}{\lambda} - \nabla\Delta\phi_{a,b}^{s_0,i}]) \tag{6}$$

where an AFV closer to 1.0 represents a position in which all of the double-differenced ambiguity values would be very nearly integer.

Since we know that the ambiguity values **must** be integers, the application of this method to GPS is as simple as defining a 3D search space, calculating the AFV for each and every point (down to some pre-defined resolution) in the search space, and picking the point with the highest value. As simple as this sounds, this technique is almost never used in practice for several reasons:

1. Depending on the size of the search space and the desired resolution, the number of points that must be evaluated can become intractable.
2. The method pre-supposes all satellite observations to be error-free; however, a relatively small amount of error in one or more observation can result in an AFV that is drastically far from 1.0.
3. Unmodeled errors can result in an incorrect set of coordinates having a higher AFV than the correct baseline coordinates.
4. The method cannot be used over time unless both receivers remain stationary since we are searching in the position domain.

The reason this method is so intractable is due to the minimum resolution required to guarantee that the correct position is not missed. Since we are searching for positions that make the double-differenced ambiguities the most nearly integer, we must actually evaluate a position that is quite close to the intersecting ambiguity lines in the first place. It is clear, for example, that at a search resolution of 9.5 cm, or one-half the carrier wavelength, it is possible for the set of evaluated locations (in one dimension) to come no closer to the correct integer gridline than 4.75 cm, as shown in Figure 2.

By plugging this worst-case residual value of 4.75 cm into AFV Equation 6, we see this would result in an AFV of 0.0, clearly nowhere near 1.0. In order to achieve a reasonable AFV of at least 0.8, for example, the absolute minimum

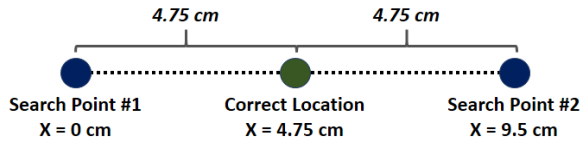


Figure 2: Maximum 1D AFM error due to a search resolution of 9.5 cm

search resolution required would be nearly 4 cm. Given a typical GPS 3D RMS position error of 4 m, this would require $(\frac{4\ m}{4\ cm})^3 = 1,000,000$ individual search points for a single observation epoch!

In addition to the computational complexity of searching through the entire search space in the absence of measurement noise, real-world errors dictate that either much higher resolutions or much larger search spaces be used to ensure that the correct position is not missed. As such, AFM has been all but abandoned for use in modern GPS localization techniques; however, it should be noted that this method alone stands apart from the rest due to its complete immunity to cycle slips. The fact that the method simply takes the integerness of a solution into account coupled with the fact that cycle slips are always integer in nature (barring half-cycle slips which will be discussed in 4.3) means that cycle slips which normally wreak havoc on other positioning techniques pose no problems whatsoever for the Ambiguity Function Method.

4.2 AFV Peak Tracking (APT) Solution

Because of the AFM's immunity to cycle slips, as well as its direct position-domain search space, it was chosen as the preferred method on which to build our baseline determination algorithm. Unfortunately, due to the errors and complications listed in the previous section, it could not be coupled with our tracking results and simply applied as a direct solution to the decimeter-scale relative localization problem.

Instead, we view the AFM search space in a different light, creating a contour-map topology in which the AFV at a specific point corresponds to the point's overall "fitness" as a potential candidate for the baseline solution. Figure 3 shows

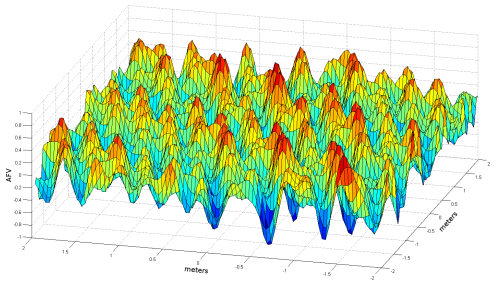


Figure 3: AFM search space as a contour map in 2D

a graphical representation of such a contour map in two dimensions, where the X and Y axes correspond to the X and Y dimensions in the standard AFM search space, and the Z axis represents the corresponding AFV, or fitness value, at the specified location. Red areas in this figure indicate regions in which the *correct* relative baseline is more likely to fall, while blue areas indicate regions of unfitness. It is clear, although the map is quite jagged and hilly, that the overall fitness function is smooth and continuous over short intervals, with easily identifiable *peaks* and *valleys* present in the topology. It is precisely these peaks that we are referring to in our name "AFV Peak Tracking Solution."

The previous section explained how AFVs close to 1.0 correspond to baselines that make the double-differenced integer ambiguities in our model the most integer-like, and therefore, the most "correct" in a mathematical sense. This becomes quite easy to see in the previous figure by recognizing that the various peaks correspond to likely baseline candidates. We also discussed that, due to multipath and unmodeled errors, there is a high probability that the highest peak will not actually correspond to the correct baseline solution. Recall from Figure 1, however, that the **correct** baseline position will experience continually low error residuals (i.e. will remain a peak) through significant changes in both satellite and receiver geometry, thanks to the accuracy of our precise receiver-receiver tracking results. Extending that concept to our AFV search grid, it is clear that peaks at incorrect locations will fade and turn into valleys as the change in geometry causes their fitnesses to decrease. As such, the "tracking" in our AFV Peak Tracking Solution refers to the fact that we not only identify the highest peak(s) at a given point in time, but also track these peaks such that it becomes possible to filter out the erroneous candidate baselines as their corresponding AFVs decrease.

In the simplest of cases, this procedure equates to initially searching over some pre-defined search space for all of the AFV peaks (i.e. baseline candidates) and then filtering through time by evaluating each remaining peak according to the newest epoch of GPS data and removing any baselines from the candidate set that are no longer valid. At some point, there will only be one valid peak remaining, corresponding to the correct relative baseline between the two receivers.

As discussed in the previous section, however, we are not carrying out this technique in the ideal situation in which we can simply ignore the unmodeled errors and assume that we are either completely stationary or that our tracking results are completely error-free. In such a case, we would simply be able to track the *values* of the peaks through time, but in our case, we are required to track the *peak locations* as well as their values, since they are most likely in constant motion, and our knowledge of these motions is not without error.

4.3 Extended AFM with Hill Climbing

Although our RegTrack algorithm provides highly accurate tracking results, they are nonetheless imperfect. As mentioned earlier, even single centimeters of error can lead to drastically low AFV levels. As such, a method of correcting for or overcoming these inaccuracies must be employed to make the tracking results useful in a motion-enabled, AFM-based technique. As such, we extend the AFM method with a simple hill climbing algorithm to overcome any inaccuracies in our tracking results and derive a new thresholding function to be used in determining an appropriate AFV level under which to filter our results.

Hill climbing algorithms provide a useful methodology for overcoming small optimization discrepancies by searching in

a region close to or around an arbitrary initial position estimate for the "local maximum" of some fitness function. In our case, the fitness function is the AFV equation itself, and the location around which to search is an estimate of the baseline between two receivers that has been tracked through time (i.e., the "tracked peak" introduced in the previous section).

Since the tracking results produced by our RegTrack algorithm are able to provide sub-centimeter-scale precision *over single epochs of time*, regardless of the relative change in position over a single epoch, the tracking errors should not be large enough to push a potential baseline candidate so far from its corresponding local AFV maximum that it enters the domain of a different (i.e., wrong) local maximum. In other words, the local maximum of the AFV function for a given position at time t will be the same local maximum for the updated position at time $t+1$ if our tracking results are used over a single epoch.

This statement is only true for a finite amount of time, however. In order to guarantee that this claim always remains true, we must re-position our baseline estimate after every time step such that it always coincides with its corresponding local maximum. In essence, this is the same as removing any bias that would normally accumulate over time in a dead-reckoning tracking algorithm such as ours.

Fortunately, since the AFV equation represents a continuous, smooth function in the position domain, we can directly use a steepest-ascent hill-climbing technique to quickly find the local maximum given our initial baseline estimate. All that this entails is evaluating the AFV equation for the points immediately around the estimate, and then repositioning it to be the point that experienced the largest increase in AFV from the original estimate. This process is repeated for the updated estimate and so on until no point is found with a better AFV than the current one. The corresponding point will be the local maximum of the function in the desired region.

At the next time step, we track any relative receiver motion using RegTrack, update our baseline estimate with the tracking results, and then carry out this hill-climbing technique to once again remove any tracking error we incurred. In such a way, any position estimate (even if it is the wrong one) can be continuously tracked through time according to its fitness within the paradigm of the AFM. There remains to be seen, however, an effective way to remove candidates that no longer fit based on the changing satellite and receiver geometries.

4.3.1 AFV Thresholding Function

We mentioned that some sort of thresholding function would be an ideal way to filter out baseline positions that have become unfit candidates over time. Such a thresholding function can be formed by taking into account that:

- Since we are searching over a discrete search space, there will almost always be some amount of error due to the offset between the "correct" position and its nearest search point in the grid, and
- The carrier phase observations used for AFV determination are not perfect measurements and will include some amount of error, due primarily in this case to multipath and receiver noise.

As such, the error term that was omitted from Equation 2 in formulation of the AFV model should not be overlooked. If we include a general catch-all error term in this equation, we are left with an AFV (corresponding to an arbitrary single satellite observation, s) of:

$$\mathrm{AFV}^s = \cos\left(2\pi \cdot \left[\frac{\nabla\Delta R_{a,b}^{s_0,s}}{\lambda} - \nabla\Delta\phi_{a,b}^{s_0,s} + \nabla\Delta\varepsilon_{a,b}^{s_0,s}\right]\right) \quad (7)$$

The unknowns in this equation are the 3D baseline coordinates (used to determine the $\nabla\Delta R_{a,b}^{s_0,s}$ term) and the $\nabla\Delta\varepsilon_{a,b}^{s_0,s}$ error term. Not immediately apparent is the fact that the λ term may also be unknown. We discussed the nature of cycle slips earlier, but some receivers also suffer from what are known as "half-cycle slips." These slips occur because many receivers choose to remove encoded navigation messages from a received GPS signal by simply squaring the received waveform. Since a squaring operation makes all negative values appear to be positive, the resulting squared carrier wave will not repeat itself every λ meters, but rather every $\frac{\lambda}{2}$ meters. Thus, a slip could potentially appear to add or remove cycles equal to only one-half the wavelength of the carrier wave.

Fortunately, most receivers that operate this way are able to detect when their carrier phase tracking loops get out of sync with the unsquared version of the received signal by a half-wavelength and will return this information along with the raw carrier phase measurement to indicate that ambiguity resolution *may* need to be carried out using half-wavelengths, denoted from here on as $\lambda_{1/2}$. As such, the λ value in Equation 7 must be treated as a pseudo-variable since its value can change and the AFV equation depends on it, but it will always be known prior to use.

Finally, it was previously discussed that the AFV equation will necessarily contain some amount of error due to the resolution of the search space. We gave an example in one-dimension that showed the worst-case error to be equal to one-half the search resolution ($\frac{res}{2}$). However, GPS operates in three dimensions, so it is possible for the search point location (which determines the calculated $\nabla\Delta R$ value) and the correct receiver location (which determines the $\nabla\Delta\phi$ value) to differ by up to $\sqrt{(\frac{res}{2})^2 + (\frac{res}{2})^2 + (\frac{res}{2})^2} = \frac{res\sqrt{3}}{2}$.

Now that we have determined the worst-case error due to the search resolution, investigated the possibility of a half-cycle slip, and included a catch-all term for any unmodeled errors (dominated by multipath), we can re-write the AFV Equation 7 for satellite s as a single-satellite, worst-case value, like so:

$$\mathrm{AFV}_{worst}^s(res, \lambda, \varepsilon) = \cos\left(2\pi\left[\frac{res\sqrt{3}}{2\lambda} + \varepsilon\right]\right) \quad (8)$$

Extending this to include all valid observations, n, for a given epoch, we arrive at an AFV thresholding function, dependent on the search resolution used, ambiguity resolution required, and worst-case error present in any given observation:

$$\mathrm{AFV}_{th}(res, \lambda, \varepsilon) = \frac{1}{n}\sum_{i=1}^{n} \cos\left(2\pi\left[\frac{res\sqrt{3}}{2\lambda} + \varepsilon\right]\right) \quad (9)$$

Algorithm 1 AFV Peak Tracking (APT) Solution

Initialization

1: Evaluate the AFV for all 3D grid coordinates in a given search cube
2: Identify all peaks from the resulting AFV set

Calibration

3: Wait until the next epoch of data arrives
4: **for** each identified peak **do**
5: Update the peak location according to the computed tracking result
6: Re-evaluate the AFV at the updated peak location
7: Hill climb by steepest ascent to the local peak maximum
8: **end for**
9: Calculate the worst-case AFV threshold value for the current epoch using Equation 9
10: Filter the updated peak locations based on the computed threshold value
11: If more than one peak remains, go to Step 3

Steady-State Localization

12: Update the peak location according to the computed tracking result
13: Re-evaluate the AFV and hill climb to the local peak maximum
14: **if** AFV remains at an acceptable level **then**
15: Go to Step 12 at the next time epoch
16: **else**
17: Go to Step 1
18: **end if**

Note in this equation that the λ value may be different for individual satellite measurements since not all observations require half-cycle ambiguity resolution. Also note that carrier phase errors due to multipath have a maximum error of ~5 cm [2], but not all observations will experience multipath, and those that do are not likely to experience it in the same direction or with the same magnitude as one another; therefore, a reasonable technique is to scale down the maximum value of multipath error (ε_M^{max}) by the number of visible satellites (n) and use the resulting value ($\varepsilon_M^{max}/n = 0.05/n$) as the ε term in the AFV thresholding function.

4.4 Putting It All Together

This section introduced several concepts used by the APT localization algorithm, but we have yet to put them together and state them as a complete methodology for relative baseline localization through time. The explicit steps in APT can be separated into three logical phases of execution: Initialization, Calibration, and Steady-State Localization. Algorithm 1, shown above, details the explicit steps comprising each of these phases.

The goal of this algorithm is to reach the steady-state phase for every remote node participating in the localization procedure. This occurs when only one peak candidate is left for a given receiver, corresponding to the correct relative baseline between the receiver carrying out the localization procedure and the remote node. Once this stage is reached, the relative baseline between the two receivers (i.e. the current peak location) can be computed at a high level of accuracy using only minimal computational resources.

It should be noted that since we have a set of potential relative coordinates at each time step which, by definition, correspond to sets of integer ambiguities containing minimal error, it is trivial to actually calculate the integer ambiguities for each peak using quite literally any satellite as a reference and then to store them for later use. At the next time step, if the satellite previously used as a reference is still visible and without error, the integer ambiguities can be used to compute the current relative baseline in a least-squares sense, thereby providing a means of both verifying the validity of the location of the tracked peaks (i.e. a sanity check) and also providing additional criteria on which to filter and remove unfit peak candidates.

5 Implementation

In order to experimentally verify the validity of the new algorithms and methodologies described in this paper, we decided to create a fully functional proof of concept. This section describes the system and implementation details of the working prototype.

5.1 Software

All software was written in pure Java, ensuring cross-compatibility between various systems and allowing for rapid development of an Android-based test application. Figure 4 outlines the individual software components comprising our relative localization system and shows how they connect and interact with one another. The solid black line labeled "Framework" delineates a standalone, platform-independent software service that can be instantiated any number of times on any number of devices without the device or user knowing anything about the inner workings of the framework. The interfaces into and out of the framework are denoted by the yellow rectangles connected to the outside of the framework boundary. These interfaces are implementation-specific, depending on the such parameters as the data format of the GPS chip used for testing and the type of networking technology being employed (e.g., Bluetooth, 3G, UDP Multicast, etc.).

The framework can currently run as both a PC service on any operating system or on any handheld device running Android version 4.0 or later. This proved to be very useful in creating and analyzing experimental results since it allowed us to conduct experiments very early in the project on both mobile and laptop devices and then continue to use the raw log files from these experiments in further research, knowing that the computational results of the research would be identical across devices, as only one code base was used.

5.2 Hardware

An actual working prototype of the distributed system shown in Figure 4 was implemented using a network of Android smartphones and tablets. Specifically, three HTC De-

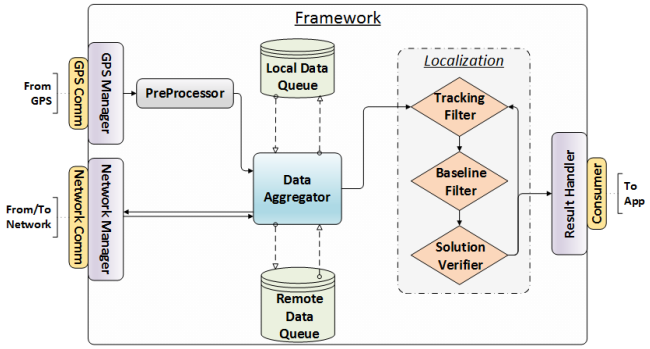


Figure 4: Software framework functional diagram

sire smartphones and three Google Nexus 7 tablets provided us a network of six devices with varying levels of processing power and memory capabilities on which to test our system. Each device was paired with exactly one custom Bluetooth-enabled sensor node [20] that included a μBlox LEA-6T GPS receiver. We chose this particular L1 receiver because it supplies raw measurement data, whereas many current low-cost receivers do not. The standalone accuracy of the LEA-6T is around 2.5 m with an unobstructed view of the sky [18].

In our setup, all GPS measurements were obtained via an external antenna connected directly to the Bluetooth sensor node which streams raw GPS data (pseudorange, carrier phase, ephemerides, etc.) over a virtual COM port to its paired Android device using the UBX proprietary protocol of the μBlox GPS receiver. The GPS coordinates computed and reported by the receiver were also streamed to the device to allow for comparison between our methodology and the built-in algorithms supplied by μBlox.

To avoid having a single point of failure, we opted for a distributed, symmetric architecture whereby each device shares its raw GPS data with the entire network and runs the localization algorithm independently on the data received from its peers, as well as from the local GPS receiver. We relied on 3G data communications and an Amazon-based cloud server for network-wide broadcast of the raw GPS readings. In our prototype, we configured the framework to use UDP over WiFi/3G to send data to the centralized cloud server whose purpose was to keep track of the participating GPS receivers and to re-broadcast any received data packets to all other networked nodes.

In playback mode (i.e. when pre-recorded log files were played using a PC-version of the framework), multiple instances of the framework were instantiated in separate processes to simulate the behavior of the framework running on physically unique devices. Each instance communicated with all other local instances via UDP-based loopback multicast over the PC's network interface. In other words, from the framework's point of view, each instance ran on its own dedicated CPU and communicated via a physical network link.

6 Evaluation

In order to evaluate our system, we devised a set of experiments to test the accuracy of our methodology in a variety of different environments, with varying levels of motion, and with significantly differing baseline lengths. We compared our results to those produced directly by the μBlox chips which implement an Augmented GPS (GPS+WAAS) solution in order to give a reasonable estimate of the relative increase in precision when our technique is compared to that of a well-established and favorably-regarded GPS chip manufacturer.

To put the results into context, another methodology was implemented to transform the *absolute* coordinates reported by the GPS chips into the *relative* coordinate system used by our localization procedures. To achieve this, we simply subtracted the absolute positions of one or more receivers from the absolute position of the local receiver currently being used as a reference (the receiver carrying out the localization and tracking operations), which should, it itself, improve the accuracy of the μBlox results by removing any correlated errors from the reported positions. In this way, it became possible to create a map of relative locations with respect to the local receiver, exactly as they appear in our own methodology. Results obtained through use of the APT algorithm introduced in this paper will be referred to simply as "APT" results. Results used for comparison and obtained from the built-in methods of the μBlox GPS chip will be referred to as "μBlox" results.

In all of the experiments in this section, the initial search space was defined to be an 8 m^3 cube, centered on the estimated baseline reported by the GPS receivers themselves. Note that in several experiments, the error in the reported μBlox baseline caused the correct solution to lie *outside* of the 8 m^3 search space. In these cases, an arbitrary initialization point was selected inside of the search space for purposes of evaluating the system; however, in real-world scenarios, either a larger initial search space must be used or improved initial baseline determination methods must be employed (note that many such methods exist, they were simply not utilized).

6.1 Stationary Setup

This setup comprised two different experiments meant to test our APT localization algorithm in the simplest and most benign of cases. Since no motion took place, the results are useful for indicating the worst-case calibration times of the system. Likewise, since multipath and cycle slips are unlikely to occur, this represents a benign scenario in terms of dealing with uncorrectable errors.

Our first experiment involved placing three colocated nodes in a wide open field unlikely to be affected by multipath. The algorithm was simply left to run for 10 minutes, producing the following steady-state results in terms of baseline deviations from the known range between pairs of individual nodes:

Node Pair	Mean Error	Standard Deviation	Calibration Time
#1 and #2	5.2 mm	2.2 mm	272 s
#1 and #3	5.8 mm	2.6 mm	327 s
#2 and #3	4.2 mm	1.8 mm	309 s
TOTAL:	5.1 mm	2.2 mm	—

Table 1: Mean errors and calibration times for baseline solution of stationary node pairs

The incredibly low error residuals for all node pairs indicate that the initial baseline calibration was successful and remained steady through time. Even more importantly, the standard deviations of the errors indicate that the hill-climbing algorithm is successfully removing any tracking bias from the solution before it can accumulate into a measurable amount of baseline error. Note that the calibration times from all nodes in this experiment average 303 s. This value is indicative of the worst-case calibration time that would ever be experienced by the system, since our nodes are both colocated and stationary, meaning that all geometry changes in the system are due exclusively to the motion of the satellites.

An additional stationary experiment was carried out on the field portion of a football stadium to test the accuracy of our system when the baseline is not on the order of centimeters. In this case, two nodes were placed at the very bottom of the bleachers on either side of the 50-yard line, approximately 221.5 feet (67.5 meters) apart. In addition to providing a test case using a longer baseline, the stadium itself provided many more opportunities for multipath and blockage of satellite visibility; nonetheless, this can still be considered a relatively benign GPS environment.

The algorithm was allowed to run for 30 minutes, producing an average error of only 6 mm with a standard deviation of 4 mm. As with the previous experiment, this indicates a successful test with a large increase in accuracy over the μBlox solution which incurred an average error of 90.2 cm with a standard deviation of 17.2 cm.

6.2 Fixed 9-foot Distance

For this experiment, we reused logs from one of the tracking experiments used to test the RegTrack methodology in [7]. This gave us the ability to compare our current technique with the dead-reckoning, tracking-only solution presented in that paper.

The setup for this experiment consisted of a stationary node positioned at one corner of a standard running track and two mobile nodes connected to a fixed 9-foot pole moving around the track in a configuration perpendicular to the running lanes. This experiment is useful because the relative distances of the nodes attached to the poles are precisely known at all times, with only the relative directions changing. As such, they provide us with a way to measure the accuracy of our technique in the "free mobility" case with extreme precision. The additional static node adds range mobility to the experiment, since both the relative directions **and** distances to the nodes are changing continuously when viewed from this receiver. This provides performance statistics on the technique when used with baselines of varying lengths.

The following graph, Figure 5, shows the estimated baseline between the mobile receivers traversing a lap around the running track as seen by the stationary reference node after its successful calibration of the relative baselines to both of the remote nodes. Figure 6 shows the same results as computed pairwise directly between the two mobile nodes.

In both cases, the results from this experiment indicate a similar level of error to that of the RegTrack experiment, although the standard deviations of the error are approximately doubled. Specifically, the mean error for the relative baseline as seen from the stationary reference node was 26.2 cm with a

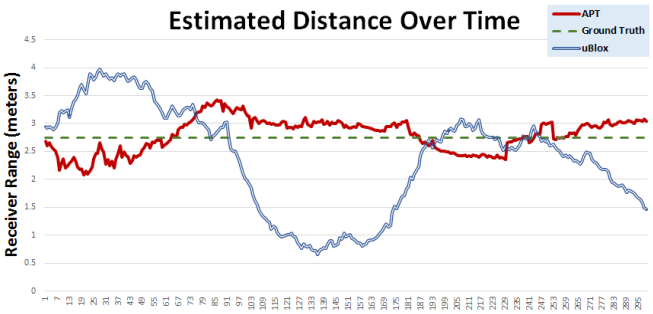


Figure 5: Estimated distance between two mobile nodes (as seen by a stationary node) as a function of time as they moved around the track

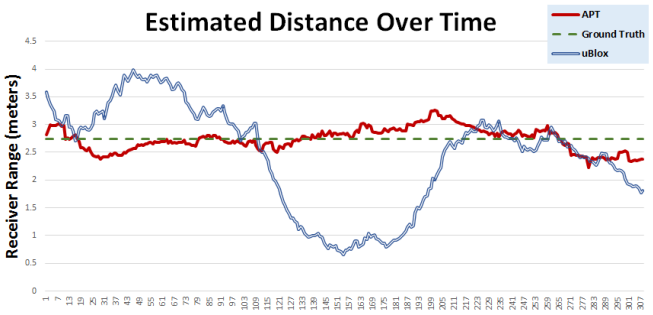


Figure 6: Estimated direct distance between two mobile nodes as a function of time as they moved around the track

standard deviation of 15.3 cm. The mean error for the direct baseline solution was 16.9 cm with a standard deviation of 12.3 cm. While these results indicate a significant improvement over the μBlox average of 89 cm of error, we would have expected to see an improvement over the dead-reckoning results from the tracking-only solution as well.

One reason for the slight decrease in accuracy and stability is an overly pessimistic view of the current baseline solution by the APT filter due to arbitrary regions of abnormally large tracking errors noted in the RegTrack solution. The subcentimeter results from that technique seem quite good, especially given the cumulative tracking errors described in [7] which indicate that the majority of *single-epoch* errors must be close to zero mean when aggregated over time; however, an analysis of the individual epoch-by-epoch results indicates that the total error distribution is not Gaussian. In fact, very clear *regions* of decimeter-scale error can be discerned amidst a vast number of results containing only millimeters of single-epoch tracking error.

There are four causes for these types of anomalies:

1. The number of non-erroneous satellite measurements as determined by the tracking algorithm has reached a critical level of 4, meaning the system is no longer over-determined,
2. The number of satellite observations is less than 4, meaning the tracking results are being determined by a motion model instead of the tracking algorithm itself,
3. One or more undetected cycle slips were present in the tracking observations, or

4. The level of multipath during the error-prone time frame was large enough (and affected enough observations) that it translated into a comparable tracking error.

These regions of error often manifest with a noise-like quality (i.e. they tend to be close to zero mean over the long term), and thus, have minimal impact on the tracking algorithm when used in a dead-reckoning framework; however, it is possible for them to be problematic in the APT localization algorithm if not detected and mitigated as much as possible, as we demonstrated that even very small tracking inaccuracies can result in a low AFV. Fortunately, the first two causes of inaccuracy can be determined by the tracking algorithm itself and passed to the APT algorithm for special handling. The latter two causes, however, are currently undetectable and will require additional processing and error handling in subsequent baseline localization procedures.

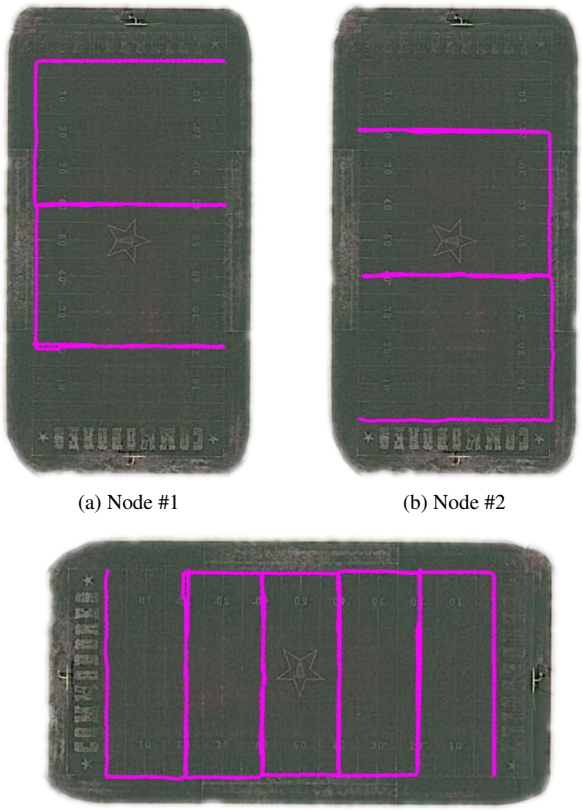
In any case, these errors sometimes cause our lock on the peak corresponding to the current baseline solution to be lost, such that the filter begins tracking a nearby peak with a much lower AFV than the "correct" peak. In this case, the peak will quickly fade below the cutoff threshold, causing the filter to re-enter its calibration stage to search for new peak candidates. Since the most recently tracked peak will have corresponded to an incorrect baseline, the error will be proportionally larger during both the erroneous steady-state and subsequent calibration phases until a new, correct peak is identified.

Even with these regions of uncertainty, the relative localization procedure provides a 6-7x improvement over the μBlox solution, which is clearly visible in the previous two figures. Additionally, the algorithm's ability to re-initialize itself on the fly provides a valuable asset for those situations when tracking alone becomes unreliable.

The times spent in the calibration phase for this experiment were significantly shorter than in the stationary case, as expected. The transition from calibration to steady-state localization in this experiment when viewed from the point of view of the stationary node was 38 s, which as a frame of reference, occurred after only 12.64 m of receiver motion. When the baseline was computed directly by the two nodes in motion, the steady-state phase was reached after 72 s of calibration. The reason for the longer calibration time in the direct baseline solution is most likely due to the simple fact that only the baseline *directionality* changes in this case, whereas both the directionality and the *range* are changing from the point of view of the reference node. In either case, these calibration times on the order of a minute indicate that this method may find utility in applications requiring rapid relative position determination.

6.3 Long Baselines

Since all experiments using short baselines and relatively slow motions showed an increase in precision over the μBlox results, we decided to test our algorithm using longer baselines with similar levels of mobility. In this experiment, we again utilized the football field from the stationary experiments; however, we also set up a stationary node on the roof of a building 1.42 km away, giving us the ability to create absolute ground truth maps for easier visualization. Two mobile nodes were initially placed on the sidelines of the football field: one at the 0-yard line and one at the 20-yard line. The nodes were walked in parallel along their respective yard lines to the opposite sideline, at which point the 0-yard line receiver walked along the sideline to the 40-yard line. Both nodes again walked in parallel to the opposite sideline, whereupon the 20-yard-line node walked until it reached the 60-yard line. This laddering process was repeated until the entire football field had been traversed.



(a) Node #1 (b) Node #2

(c) Composite Tracks

Figure 7: Tracks of two nodes moving in parallel ~20 yards apart on a football field

Figure 7 shows the tracks of the mobile nodes in this experiment as calculated by the receiver on a building almost a mile away. From these figures, it is clear that the increased baseline length between the receiver performing the localization and the remote nodes has negligible impact on the accuracy of the results. In fact, the paths traversed by each of the mobile receivers are obvious from these graphics alone.

For comparison, Figure 8 shows a portion of our APT results (in pink) overlayed by the absolute track produced by one of the μBlox nodes (in yellow), where the ground truth is precisely the 40-yard line:

Figure 8: Localization comparison of the absolute μBlox solution (yellow) to the relative APT solution (pink)

Note that this is a strictly qualitative comparison in which the **absolute** μBlox results are compared to the **relative** APT results produced in relation to a stationary node 1.42 km away. This figure shows both the obvious increase in precision produced by our technique, as well as a substantial increase in stability through time. The μBlox variations in this figure of up to $\sim$3 yards indicate absolute errors from the ground truth of over 2.5 m.

6.4 Driving

For the following set of experiments, we re-used the data from a driving setup from our earlier work in [7] in order to compare the results from our current technique to the tracking-only results presented in that paper. Our setup included three mobile nodes mounted to the roof of a car with a single stationary node placed in a parking lot at the starting site of the driving trial. The primary purpose of these experiments was to test the accuracy of our methodology under high dynamic situations and at increasing baseline lengths.

6.4.1 Driving in an Obstructed Area

The driving course was separated into two distinct parts, the first of which included driving in multipath-rich alleyways and on suburban roads for a total of 1.48 km. In the case that the relative baseline was determined using a stationary reference node, the results from the APT solution algorithm were an order of 2 times *less* accurate than those from the tracking-only solution reported in [7], but 3 times *more* accurate than the μBlox results. On the other hand, direct localization produced results that had similar or even slightly better accuracies than the tracking-only solution, with cumulative errors as shown in Table 2 for one of the pairs of remote nodes over increasing intervals of 500 m.

Method	Distance	Mean Error	Std Dev
Reference	500 m	39.1 cm	25.5 cm
Direct	500 m	58.3 cm	28.7 cm
μBlox	500 m	93.6 cm	36.5 cm
Reference	1 km	34.7 cm	25.2 cm
Direct	1 km	34.0 cm	32.6 cm
μBlox	1 km	97.5 cm	36.5 cm
Reference	1.48 km	36.5 cm	23.4 cm
Direct	1.48 km	39.7 cm	37.5 cm
μBlox	1.48 km	113.4 cm	55.6 cm

Table 2: Cumulative localization accuracy for one node pair driving through a difficult GPS environment. "Reference" results indicate baselines computed via subtraction of the mobile node ranges with respect to a stationary reference node, and "Direct" results indicate baselines computed pairwise directly between the mobile nodes themselves.

These results indicate a quantifiable improvement over the μBlox algorithms, but an unexpected lack of substantial increase in accuracy from the tracking-only results in [7]. The number of filter re-initializations was higher than expected and corresponded most commonly to the times when the vehicle had a severely obstructed view of the sky, or in other words, when driving very close to buildings in a narrow alleyway or under leafy trees. As such, it appears that multipath plays a much greater role in the ability of our APT baseline filter to maintain a lock on the correct peak than initially anticipated. Nonetheless, this portion of the experiment ended with all three of the remote nodes in a configuration containing less than 1 meter of error each from the correct ground truth position.

The transition times from calibration to the steady-state phase of the APT localization algorithm were 31 s, 25 s, and 50 s respectively for baseline determination of each of the mobile nodes as calculated by the stationary reference. The average calibration time for direct baseline determination between the roving nodes themselves was 90 s, almost 3 times as long as the average for the reference node. Again, the reason for this dichotomy in calibration times is due to the fact that the baseline *ranges* between the stationary node and the vehicle nodes are changing quite rapidly, whereas the ranges between the vehicle nodes themselves are not changing at all. The decrease in time spent in the calibration phase for the stationary receiver is consistent with our expectation that increased relative motion will result in decreased calibration times.

6.4.2 High-Speed Driving

At this point, we turned onto an interstate highway and began driving at an average speed of 90 km/h over an additional 7 km of road. For the duration of this experiment, all receivers had consistently clear views of the entire sky (with the exception of an occasional overpass). As such, we expected relatively few tracking errors due to multipath and a correspondingly low number of APT filter re-initializations.

Compare the errors from our APT technique to those obtained from the μBlox chip in Figure 9 and the increased precision of our localization technique becomes apparent, especially in terms of the baseline between nodes #2 and #3 in which the instantaneous localization errors rarely exceed 50 cm. Additionally, the number of times when the accuracy of the μBlox solution over multiple epochs is better than our own is very small, most notably at the very end of the experiment for nodes #1 and #2.

As with the tracking-only results, this increase in accuracy over the μBlox results is substantial, and we believe the accuracy could have been even higher if not for the overpasses, which almost always caused the localization filter to require re-initialization. In fact, many of the regions of decreased accuracy between nodes #1 and #2 began at times corresponding to the vehicle passing underneath an overpass (note that error spikes in the graph for nodes #2 and #3 seem to be more random, having little correlation with environmental obstacles at corresponding times).

For comparison, the mean range error averaged over all remote nodes on the car when viewed from a reference receiver in the parking lot was 49.9 cm with a standard deviation of 30.4 cm. This represents a 5x improvement over the μBlox solution in terms of both error residuals and stability as evinced by the standard deviation. Likewise, the mean error when the relative baselines are computed directly between the roving nodes themselves was 39.9 cm with a standard deviation of 35.6 cm, almost identical to the results obtained from the tracking-only solution.

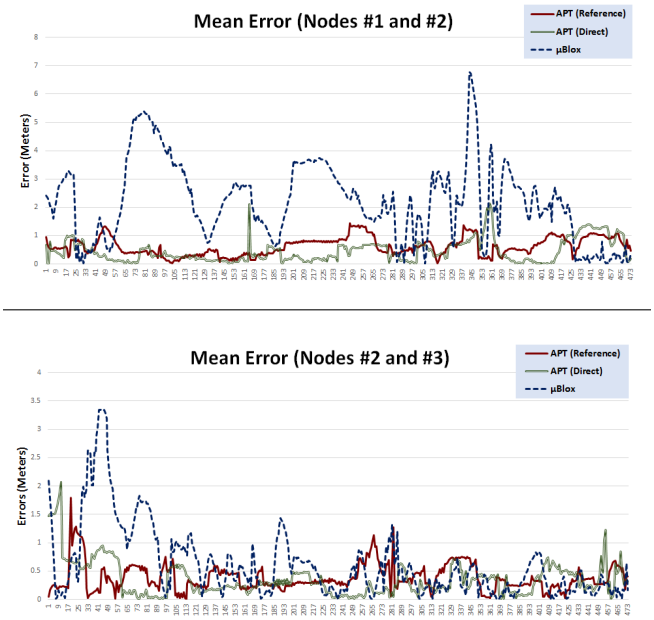


Figure 9: Mean localization errors over time for two mobile node pairs attached to a car roof driving along the interstate

6.4.3 Closing the Loop

Recall from the tracking-only results presented in [7] that we encountered a problem at the very end of the experiment, namely that we passed underneath a wide overpass while changing directions, causing incorrect tracking updates due to a momentary loss of satellite locks. One of the primary benefits of our APT localization approach is that the baseline filter quickly destabilizes as the AFV of erroneous peaks drops below a threshold value. In this experiment, this occurred when we traveled underneath the final overpass and experienced measurable amounts of tracking error. At this point, the filter successfully re-initialized itself with a better baseline solution, and we were able to continue localizing with a high degree of accuracy, unlike in prior experiments using RegTrack alone.

The following figure shows a track of the entire experiment, including a zoomed-in snapshot of the starting and ending points, indicating minimal error over the course of the entire experiment. The physical distance between the start and end points of the node track pictured below was 1.54 m; however, we failed to mark the starting point of the experiment, so it was difficult to gauge the precise location at which we should stop driving. As such, it is highly likely that a large portion of this offset was due simply to misalignment of the car at the end of the experiment.

This figure shows the usefulness and benefits of the APT baseline localization algorithm over the tracking-only solution. In cases where substantial amounts of satellite outage are likely to be experienced, the tracking solution, while accurate, will always require human intervention whenever re-initialization is required. The full APT technique, however, allows the system to re-initialize itself, thereby providing a "hands-off" automatic solution in which the system can quite literally be turned on and left to manage itself.

Figure 10: Track of the full driving experiment with emphasis on the starting and ending points of the course

7 Performance Metrics

In order to analyze the scalability and processing requirements of our system, we took each individual component in our software framework (shown in Figure 4) and analyzed the computational and memory requirements required for that component to run. Memory requirements were determined using the Eclipse Memory Analysis Toolkit, and processing latencies were computed from real-world experimentation. Results denoted by a"PC-based" benchmark were formed from the average of the corresponding operations run 10,000 times each on a series of desktop and notebook computers, and results denoted by a "mobile-based" benchmark were formed from the same operations run 10,000 times each on a Google Nexus 7 Tablet and an HTC Desire smartphone. Full details can be found in [6], with Table 3 giving a brief overview of the results.

Module Name	**Processing Time (PC)**	**Processing Time (Mobile)**
GPS Manager	6.64 μs	97.9 μs
Preprocessor	129.3 μs	4.0 ms
Network Manager	12.4 μs	405.3 μs
Data Aggregator	19.9 μs	1.08 ms
Tracking Filter	24.1 μs	845.4 μs
Baseline Filter (Calibrating)	1594.0 μs	11.5 ms
Baseline Filter (Steady-State)	850 μs	10.3 ms
Result Handler	0 μs	0 μs
TOTAL (Calibrating)	1.79 ms	17.93 ms
TOTAL (Steady-State)	1.04 ms	16.73 ms

Table 3: Summary of processing times for each software component of the localization framework

It is important to note that since our localization algorithm works on pairwise sets of satellite data, its mathematical complexity does not grow with the addition of GPS receivers to the network. In fact, from Table 3, we can deduce that our system should be able to support up to 60 steady-state mobile receivers simultaneously without running into latency issues. The caveat to this statement is that the "initializa-

tion mode" of the localization algorithm requires significantly more processing time than the nominal "steady-state localization mode;" therefore, as additional nodes enter the network, there will be intermittent periods during which position data for the new, calibrating receivers is unavailable. This is an important aspect to consider when examining the applicability of our technique to real-time systems.

Aside from the processing overhead associated with baseline calibration, however, the primary bottleneck in our system is the communication bandwidth required to broadcast the raw satellite measurements among a potentially large number of receivers. Since the amount of data transmitted from a single node every second is actually quite low (on the order of 500 bytes/second), there are better ways (than the naïve cloud-based multicast approach used in our experiments) of approaching this problem if and when communication scalability becomes an issue.

8 Future Work

One of the primary limitations of the work presented in this paper is the strong dependence of the APT localization algorithm on the initial estimate of the baseline between two receivers. The worst-case accuracy of this estimate dictates the minimum size of the search cube that must be traversed when identifying the relevant peaks and baseline candidates for a given pair of receivers. Any increase in baseline uncertainty results in a cubic increase in the number of points that must be searched to find the correct peak. Correspondingly, since the search resolution of APT is so fine, even small increases in the required size of the search space can result in drastically longer processing times, especially for the initial search procedure in which candidate peaks are identified.

In order to cut down on the time required to search through this vast number of points, additional research should be carried out on determining a better estimate of the initial baseline between two receivers before using the algorithms described in this paper. Conversely, it would also be beneficial to have a better idea of the confidence interval (or an estimate of the worst-case error) for any given baseline. Better accuracy estimates would ensure that the correct baseline is not missed in the initial peak determination algorithm, as well as minimize the number of extra points that must be evaluated in the case of an overly pessimistic accuracy estimate.

Experiments using the complete APT localization framework highlighted several drawbacks of the hill-climbing technique used to remove accumulated bias from the requisite tracking results. These include a strong dependence on the accuracy of the tracking results themselves, as well as an increased susceptibility to errors caused by multipath. As such, additional research into methods of removing short-term tracking biases should be carried out to ensure that the locality of peaks being tracked by the APT filter remains constant through time.

It would, of course, be beneficial for future applications to develop "smarter" peak searching algorithms, such that only a subset of the search space need be traversed to identify any relevant peaks. This would drastically cut down on the processing time required to initially identify the peaks, as well as the time it takes to track them. As such, the scalability of this research would improve to allow for a larger number of receivers to be localized in real-time without experiencing data bottlenecks or processing latency issues.

One final future consideration is the US Government's "GPS Modernization Project" which aims to provide two additional civilian signals, designated L2C and L5, in 2018 and 2021, respectively [13]. While these signals promise to increase the achievable accuracy of GPS receivers in the future, our solution fills a notable gap in the interim. Additionally, there is nothing in our methodology that precludes it from using these additional frequencies to further enhance our resulting accuracy and increase robustness in a variety of environments. In fact, decimeter-scale precision with our current technology could turn into millimeter-scale precision given the new civilian frequencies.

9 Conclusion

In this paper, we presented a novel approach to GPS-based differential localization of mobile nodes, with the overarching goal of dramatically increasing the precision of relative 3D baseline coordinates using only low-cost, off-the-shelf GPS receivers. By allowing a network of GPS receivers to share their raw satellite measurements with one another, we were able to achieve decimeter-scale relative localization accuracy.

Unlike most GPS-based navigation solutions, our approach does not snap positions to maps or try to use a dynamic model for the motions of the receivers (other than for tracking during satellite losses of lock). Also unlike other high-precision GPS localization techniques used in applications for which our system may have utility, our approach does not require a stationary calibration phase, relying instead on a symbiotic feedback relationship between relative tracking and baseline localization to provide real-time coordinate updates.

Our technique allows any GPS receiver to become its own reference, thus negating the need for an explicit "reference station." Likewise, we employ observation models which either do not require a reference satellite or impose no limitations on which satellite must be used as a reference (or for how long). By incorporating tracking results into our baseline determination algorithms, there is no longer the need to pre-survey receiver locations prior to application deployment or for any receiver in the network to remain stationary during calibration; as such, our methodology provides a simple out-of-the-box, ready-when-deployed solution to high accuracy relative localization.

In four different low-mobility experiments, we were able to achieve localization accuracies several factors better than the corresponding "standard" GPS positioning techniques, depending on the test environment and the quality of the satellite observations; however, the precision was slightly degraded from that of the tracking-only solution by a factor of about 1.75, except in the case of stationary localization, where precision was actually significantly improved. Based on additional investigation into single-epoch tracking anomalies, we found that this typical decrease in accuracy was due to very specific regions of higher-than-normal tracking error which caused our APT localization filter to begin tracking incorrect peaks.

Regardless of the high dependence of our localization technique on the precision of the tracking results, our methodology showed that by allowing the filter to re-initialize itself on-the-fly, it is usually possible to reacquire a lock on the "correct" peak, thereby removing the necessity for user intervention in the case of erroneous measurements. Likewise, since initialization is handled by the localization algorithm itself, calibration becomes an *implicit* step requiring no stationary setups or *a priori* knowledge of the precise relative baselines between pairs of receivers. Likewise, the time required for the filter to converge to a high-accuracy, steady-state solution is generally less than two minutes when the receivers are in motion, and anywhere from 4 to 6 minutes when stationary.

Finally, our experiments included receiver-receiver baseline lengths ranging from 0 m all the way up to 3.5 km, with little to no impact on the precision of the results. We conclude, therefore, that our method is quite robust to changing baseline lengths, and the limiting factor may be the various mathematical assumptions regarding the similarity of satellite unit direction vectors for multiple receivers in the same geographic region, as well as the assumption that these direction vectors remain relatively constant over single epochs of receiver motion in the Temporal Double-Differencing model. As such, we anticipate similar levels of accuracy to be achievable for significantly longer baselines.

While the majority of off-the-shelf and embedded GPS receivers currently only employ simple point positioning algorithms, the number of precision devices that rely on GPS as their primary localization service is growing rapidly [14, 9]. The proliferation of smartphones and GPS-enabled mobile sensors, along with the ever-increasing availability of anywhere, anytime network coverage, is creating a substantial and practical platform for the relative positioning method described in this paper. As the price of technology continues to decline and our world becomes ever more connected, further research into highly accurate, low-cost relative positioning techniques will continue to find its way even more prominently into our everyday devices, and we hope to see the approach presented in this paper spark further research and become only one of a body of ad-hoc relative localization techniques centered around low-cost, commercial GPS receivers.

10 Acknowledgments

The research presented in this paper was supported in part by the National Science Foundation under the CNS-NeTS program (Grant #1218710), a Google Research Award, Vanderbilt University, and Project #TAMOP-4.2.2.A-11/1/KONV-2012-0073 of the European Union and the European Social Fund. Additionally, we would like to thank NSF Project Manager Thyagarajan Nandagopal for his helpful suggestions regarding our research. Finally, we are very grateful to our anonymous paper reviewers for their collective input. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

11 References

[1] C. Counselman and S. Gourevitch. Miniature interferometer terminals for earth surveying: Ambiguity and multipath with Global Positioning System. *IEEE Transactions on Geoscience and Remote Sensing*, GE-19(4), October 1981.

[2] H. Ebadi. Positioning with GPS. K.N. Toosi University of Technology, August 2000.

[3] S. L. A. B. for State of Washington Department of Natural Resources. GPS guidebook: Standards and guidelines for land surveying using Global Positioning System methods. Version 1, November 2004.

[4] Garmin. What is WAAS? About GPS, Garmin Website, 2012, http://www8.garmin.com/aboutGPS/waas.html.

[5] T. Geomatics. Network RTK. GPS Network Tutorial, 2011-2012, http://tekmon.gr/2011/03/network-rtk-2/.

[6] W. Hedgecock. Precise real-time relative localization using single-frequency GPS. Dissertation, Vanderbilt University, Nashville, 05/2014 2014.

[7] W. Hedgecock, M. Maroti, J. Sallai, P. Volgyesi, and A. Ledeczi. High-accuracy differential tracking of low-cost gps receivers. In *ACM 11th International Conference on Mobile Systems, Applications, and Services (MobiSys '13)*, Taipei, Taiwan, June 2013.

[8] W. C. Inc. Relative moving baseline software (RMBS). March 2000, http://webone.novatel.ca/assets/Documents/Waypoint/Reports/RelativeMovingBaselineSoftware.pdf.

[9] B. Insight. GPS and mobile handsets. Technical Report, LBS Research Series, 2012.

[10] S. Lightbody and G. Chisholm. Techniques in relative RTK GNSS positioning. White Paper, Trimble Marine Group, 2010.

[11] N. Luo. Centimetre-level relative positioning of multiple moving platforms using ambiguity constraints. In *Proceedings of the ION GPS2000 Conference, Salt Lake City, UT*, September 2000.

[12] C. Mekik and M. Arslanoglu. Investigation on accuracies of real time kinematic GPS for GIS applications. *Remote Sensing Journal*, 1:22–35, March 2009.

[13] National Coordination Office for Space-Based Positioning, Navigation, and Timing. New civil signals. GPS.gov - Official U.S. Government information about GPS and related topics, April 2014, http://www.gps.gov/systems/gps/modernization/civilsignals/.

[14] K. J. O'Brien. Smartphone sales taking toll on GPS devices. New York Times, November 14, 2010, Online at http://www.nytimes.com/2010/11/15/technology/15iht-navigate.html.

[15] T. D. of Transportation. TxDOT survey manual - GPS RTK surveying. http://onlinemanuals.txdot.gov/txdotmanuals/ess/gps_rtk_surveying.htm, April 2011.

[16] U. States Geological Survey GPS Committee. USGS Global Positioning System. GPS Applications and Surveying Methods, http://water.usgs.gov/osw/gps/index.html.

[17] T. Takasu. RTKLIB: An open source program package for GNSS positioning. http://gpspp.sakura.ne.jp/rtklib/rtklib.htm, 2011.

[18] u-blox AG. Lea-6t module with precision timing. 2013, http://www.u-blox.com/de/lea-6t.html.

[19] R. van der Merwe and E. Wan. Sigma-point Kalman filters for integrated navigation. 2004.

[20] P. Völgyesi, S. Szilvási, J. Sallai, and A. Lédeczi. External smart microphone for mobile phones. In *Proceedings of the International Conference on Sensing Technology*, ICST, 2011.

[21] E. A. Wan and R. van der Merwe. Chapter 7. the unscented kalman filter. *Kalman Filtering and Neural Networks*, 2001.

[22] O. J. Woodman. An introduction to inertial navigation. Technical Report Number 696, University of Cambridge, August 2007.