SELF-CORRECTING ADAPTIVE TRACKING SYSTEM

Richard Lau, HeeChang Kim, Marcus Pang, Arnie Neidhardt, Arturo Cisneros, Vikram Kaul
Telcordia Technologies, Inc.

ABSTRACT

The benefits of Global Positioning System (GPS) are recognized in numerous military as well as civilian applications. In many situations, however, GPS signals are simply not available or at the best intermittently observable. This paper describes a novel approach, called Self-correcting Adaptive Tracking System (SATS), which focuses on solving group location problem when GPS is not available. In our approach, we use a tracking mechanism that allows locating group members based on their pair wise distance information. A key innovation of SATS is that we use an adaptive search algorithm to find the new position estimate based on constraints given by the ranged data. In additional, our approach is capable of extracting directional information normally unavailable in ranging system, which allows us to adaptively stabilize the orientation of the group. The SATS methodology has been prototyped and tested as part of an Office of Navy Research (ONR) program [1].

I. INTRODUCTION

Global Positioning System (GPS) is a valuable tool for self locating as well as tracking members of a group. The benefits of GPS are recognized in numerous military as well as civilian applications. In many situations, however, GPS signals are simply not available or at the best intermittently observable. This paper describes a novel approach, called Self-correcting Adaptive Tracking System (SATS), which focuses on solving group location problem when GPS is not available.

In certain applications, especially military or emergence rescue team applications, it is desirable for a team member to track the locations of his/her peer as well as a centralized location tracking the location of the team. Examples include a Marine fire team in a combat operation, or a rescue team entering a building or an underground cave. In these cases, extended absence of GPS is expected, which makes GPS-only tracking system almost unusable. The innovation described in this paper proposes a novel idea that allows such group navigation and remote tracking possible.

One traditional approach to this problem relies on an Inertia Measurement Units (IMU) and applies appropriate reference frame transformation to estimate position. However, for low-cost commercial grade IMU, the bias in the accelerometer and gyroscope of the IMU produces substantial errors and renders this approach ineffective. To stabilize error zero-velocity updating techniques have been proposed [2], where an IMU is placed at the heal of the foot or mounted at the shoe. The IMU signal is analyzed and the zero-velocity points are detected and used to correct the IMU drifting error. The ZUPTING technique theoretically can reduce the position error from the square root of time to linear of time, but is subject to errors due to inaccuracy of zero velocity detection, false positives and negatives, as well as implementation and deployment complexity. Moreover, it is applicable in the foot-walking environment, rather than running, or other movements (e.g. vehicular).

Adding ranging sensors among group members using RF or ultrasonic devices has also been studied, these techniques may provide sufficient range accuracy, but were not able to provide accurate group navigation due to lack of directional information.
In our SATS approach, instead of tracking individuals, we measure relative distances among a group of users (called cliché), and use a mathematical search algorithm to find the best location of each member. A useful feature of this approach is that orientation information about the clique can be derived and used to improve overall accuracy.

II. SATS ENVIRONMENT

The Self-correcting Adaptive Tracking System (SATS) environment is shown in Figure 1. The overall system consists of multiple *cliques* of *SATS devices*, one to three *relays*, and an *anchored station*.

Refer to Figure 1, the SATS environment includes both GPS-enabled and GPS-denied partitions. The SATS system includes the following components:

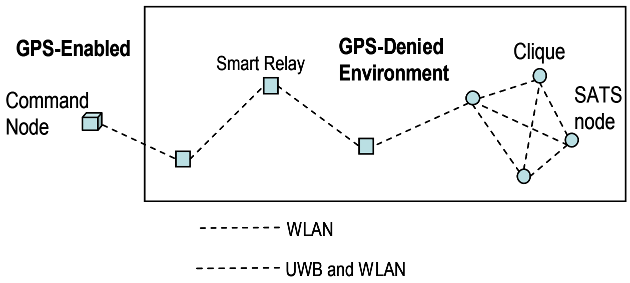

Figure 1: Navigation in a GPS-Denied Environment

- **Command Node** This is a computer that has an 802.11 interface. The Command Node is responsible for the following functions:
  - Initialization of the SATS nodes, including assignment of one SATS node (node #1) as the leading node
  - A GUI running in the North-facing mode to keep updating on the location of the clique.
  - Maintain communication with the leading SATS node via WLAN.

- **SATS Node** A number (3-20) of SATS nodes forms a clique (small fire team) and the clique has a leader SATS node. Each SATS node communicates with each other via two RF links: Ultra-WideBand (UWB) and WLAN. UWB is for pair wise ranging among SATS nodes and WLAN is used to convey other data such as IMU data. The WLAN is also used to communicate between the SATS leader node and the Command Node via Smart Relays. Each SATS node has a GUI that shows the location of clique members. Two modes of viewing are supported: 1) the forward-facing mode provides a view following the facing direction of the user (Marine), and 2) the north-facing mode provides a view that always points in the north direction (like that of a map). The north-facing mode view is the only view supported in the command station. Other auxiliary functions of the SATS node include:

  – Warning of foreign WLAN detected
  – Alert of dropping of Smart Relay
  – Icon to display the north direction (like a compass)
  – Text capability

- **Smart Relay** When a clique wanders too far from the Command Node, the SATS uses Smart Relays to relay messages between the SATS leader node and the Command Node. The Smart Relay operates on a WLAN interface. When the SATS leader node is getting far from the Command Node, it senses that the WLAN signal is weak. It then displays a message (and a beeping sound) to alert the SATS leader to drop a Smart Relay. The smart relay is a minicomputer with a WLAN interface. Once a smart relay is deployed (dropped and placed on a convenient location), it relays messages by re-broadcasting. The messages have sequence number, so that duplicated messages will be dropped by the receiver.

III. PEFFORMANCE CRITERIA

We suggest two key metrics for analysis and quantification of the performance of GPS-Denied navigation. The first is the **Relative Position Error (RPE)**. The RPE is represented by two components, the relative distance error (RDE) measured in meters and relative orientation error (ROE) measured in degrees of an angle. The second performance metric we suggest is **Clique Drift Error (CDE).** This measures the absolute error between the centroid of the polygon formed by the clique and the centroid of the SATS polygon, with respect to a fixed reference point.

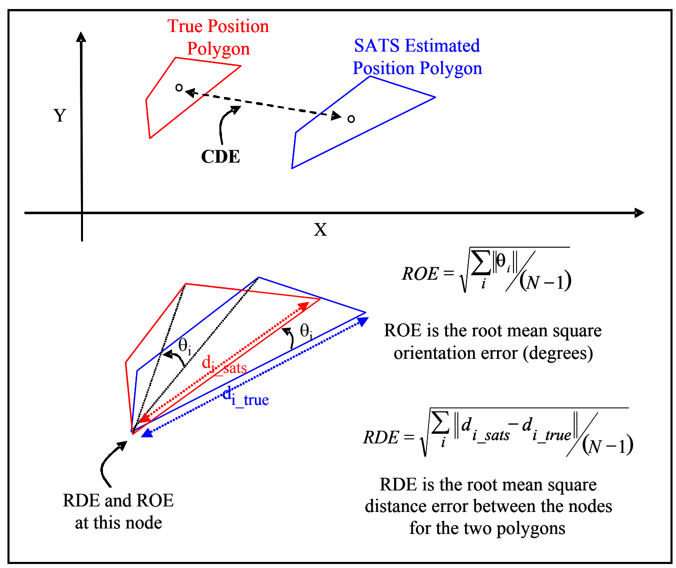

Figure 2: CDE, RPE (RDE & ROE) definition

The Precise definition of the performance metrics is:

1. Relative Position Error (RPE): RPE consists of two sub categories both dealing with relative localization of the clique:
   – Relative Distance Error (RDE): RDE is defined as the root mean square error between the SATS estimated distances and the true distances of all the nodes.
   – Relative Orientation Error (ROE): ROE is defined as the root mean square orientation error (in degrees) between the SATS estimated

orientation and the true orientation of all the nodes.

2. Clique Drift Error (CDE): CDE is defined as the absolute root mean square distance error with respect to a fixed reference point between the SATS estimated position and the true position.

In certain applications, the RPE is considered to be a more important metrics. The key strength of OPMA is targeted improving the performance of RPE.

IV. THE SATS APPROACH

We propose a novel proposal based on the assumption that the clique moves within close proximity most of the time. Each node has the following capabilities:

- Knowledge of where the node is facing (or heading)
- Knowledge of all the pair wise distances among all the nodes
- Ability to exchange information among nodes
- Ability to exchange information with a remote command node

Our novel approach is based on using UWB ranging information as a constraint tracking process. The SATS algorithm performs a search and optimization procedure to find the best new locations subject to constraints provided by updated ranging information and per-node magnetometer information. In addition, SATS has an orientation error tracking feedback loop to perform self-correction of clique level orientation errors.

IV.1 POLYGON MATCHING ALGORITHM

We shall illustrate our approach using a 4-node clique. Suppose at a time t, the estimated position of the nodes are known, we want to estimate the node positions at time t+1, each unit of time corresponds to a position estimate. Therefore, in practice each time unit may be within 1 to a few seconds. To perform the estimate of new position at t+1, we use an algorithm we call Extended Polygon Matching Algorithm (PMA). The PMA is composed of two distinct steps.

The first one is called PMA-Search and the second part is called PMA-Rotate. The goal of the PMA-Search is to find a best reference polygon whose links are closest to those of the UWB Polygon at time t+1 (U_{T+1}). As shown in Figure 2, a forward facing vector (FFV) is obtained at each node at time t, using the magnetometer. The FFV provider the estimated direction of a node between time t and t+1. Once the FFV is decided, the search positions are defined to be equidistance points on the FFV, as shown as black dots in Figure 2. These are the search positions with respect to the reference polygon. For 4 nodes, there are a total of 7^4 search points. The search criterion is given by:

$$J_s = \min_{i,j,k,l} \sum_{Poly\ Links} | R_{xy} - U_{xy} | \qquad (1)$$

where i,j,k,l ranges from 0 to $6v$, (v is a search step). xy's are the links of the polygon.
R_{xy} is the link of the reference polygon, U_{xy} is the corresponding link from UWB polygon.

Equation (1) searches for the reference polygon that is closest to the UWB polygon. Due to various errors from UWB, previous estimates, and FFV, the reference polygon will not match the UWB polygon. The second part of PMA, i.e. the PMA-Rotate is to estimate the new position at time t+1 using the UWB polygon. This is achieved by first placing the floating UWB polygon with its centroid coinciding with that of the reference polygon obtained from PMA-Search. The distances from the UWB vertices (Vu) to the reference polygon vertices (Vr) are then computed and added together to form a new penalty function with respect to an initial UWB polygon orientation of β. The UWB polygon will be rotated until the minimum of the penalty function (J_r) is found according to:

$$Jr = \min_{\beta} \sum_{i} | Vr(i,\beta) - Vu(i,\beta) | \qquad (2)$$

The positions corresponding to this minimum penalty J_r are the new clique estimate.

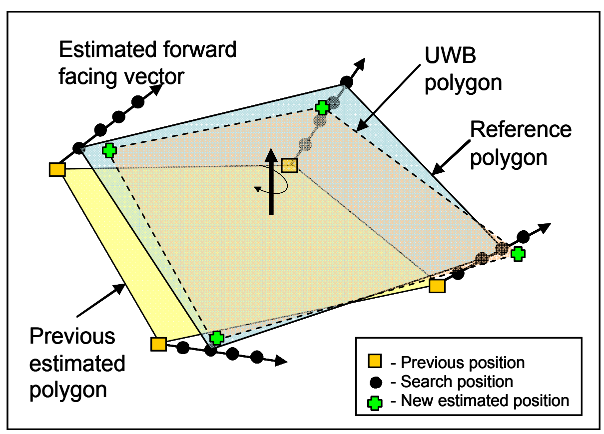

Figure 3: Polygon Matching Algorithm

IV.2 SCABLE POLYGON MATCHING ALGORITHM

The PMA-Search procedure requires N^S search steps (where N is the number of nodes and S is the number of search points). The requirement becomes computational impractical for even moderate number of nodes, say 10, making the algorithm un-scalable with respect to the number of nodes. In this section, we describe a modification to the search algorithm, which achieves a linear increase in computation with respect to the number of nodes.

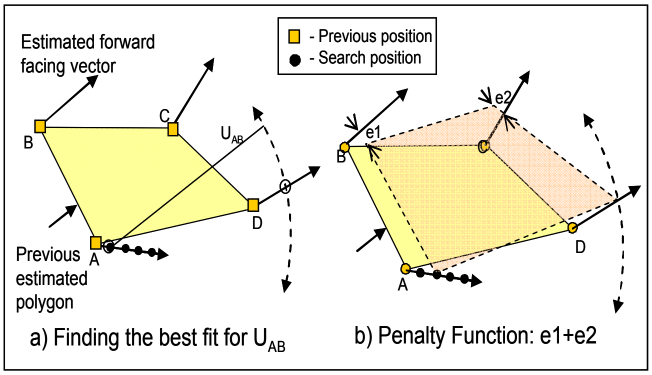

Figure 4: Scalable Polygon Matching Algorithm

Instead of searching for all the combination of the points on each of the FFV, we will just pick one node, say node A, and search on the FFV of A for S points. For each of the S points, we will draw a circle of radius given by the UWB distance between node A and node D (we select node D without loss of generality). The circle from A will intersect the FFV of D at 0,1, or 2 point. If there is no intersection, we will find the point on the circle that is closest to D's FFV, and use that as the solution point. If there are 2 solutions, we will compare the distance moved and select the one that is within the search range. Once the solution point is computed, we obtain the situation as shown in Figure 4b, in which is the new UWB polygon (dotted-line) is placed onto the search point at A and the intersection point at D. An error function of e1+e2 is then obtained, in which e1 and e2 are the perpendicular distances of the corresponding UWB vertices to B's FFV and C's FFV respectively. As the search proceeds, the minimum error function will be selected to produce the new reference polygon. Because only the S point of one node is searched, all other locations are computed, this algorithm's complexity is linear with respect to the number of nodes.

IV.3 ORIENTATION-CORRECTING PMA

The PMA described so far exerts no control on the clique orientation, which is expected to wander randomly without bound, and roughly as a linear function of time. This error accumulation could be quite bad from a user perspective. For example, if clique orientation error accumulates at a rate of 0.5^{o} per second, it would only take 6 minutes to incur 180 degrees clique orientation error. If that happens, it would mean that node A will think that node B is on its left side, while node B is really on its right side. To attain an error bound to within 45^{o} in 1 hour would require the CO error rate to be about 0.01 degree, an impractical performance benchmark.

How can we attain control over clique orientation, given that our main measurement device is distance among nodes? If we re-visit our PMA, we would notice that an assumption with respect to the PMA search was that the node reset positions are used to predict the next positions. However, the reset node positions contain errors resulted from both steps of PMA, namely, searching for the reference polygon, and the polygon matching between UWB polygon and the reference polygon. Part of this error manifests itself as the clique orientation error, which accumulates without bound. If we can somehow detect and quantify just the clique orientation error component, we will have a chance of controlling it.

As described in the last section, our capability to navigate in SATS is based on the PMA. But, can we harness more out of the PMA and allocate its capability to reduce the clique orientation error? It turns out that the PMA already provides a framework that allows measurement of clique orientation error. However, we need to modify the PMA to direct the resource for clique orientation. We will call this addition capability "Orientation-correcting PMA", or OPMA. Here is how we proceed: Find an "error function" associated with clique orientation. The error function is smallest for correct clique orientation compared to other incorrect clique orientations. We suggest using a measure of the deviation (P) between the reference polygon and the UWB polygon as the error function for measuring the degree of clique orientation error, given by:

$$P(\theta_n) = \sum_{t=t_0}^{t_0+T} \sum_i | R_i - U_i | \qquad (3)$$

where: θ_n is the clique orientation at time = t_0

R_i is i-th leg of the reference polygon resulted from PMA search

U_i is the i-th leg of the UWB polygon resulted from PMA matching

$P(\theta_n)$ is computed from t_0 to $t_0 + T$

Clique Orientation (CO) is defined as the angle between North and the a → b link

The rationale for choosing this error function is illustrated in Figure 5.

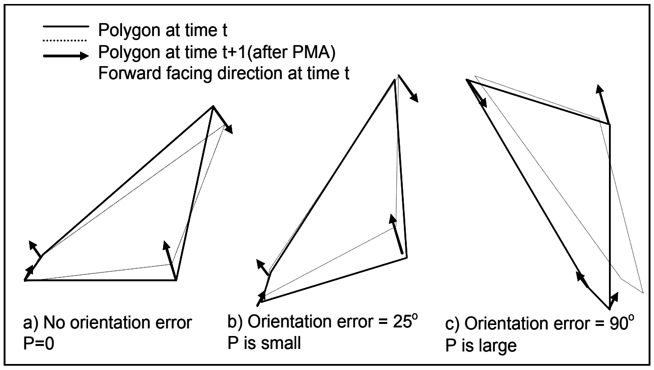

Figure 5: Clique Orientation Error Function

As shown in of Figure 5a, when the polygon has no orientation error at time t, the error function attains a value of zero, provided that there are no errors due to UBW polygon and facing vector. When the starting position of the clique has a CO of 25^{o} as shown in Figure 5b, we can see that the PMA results in a non perfect match, this error P(25^{o}) however is small. When the CO is 90^{o}, the P(90^{o}) is visually much larger. The relative small P at a CO of 25^{o} is not itself sufficient to differentiate picking the right starting CO. However, if we accumulated P for duration of T seconds, we will be able to distinguish among different starting CO's. This is why equation (1) sums the square of the difference between the reference polygon and the UWB polygon for a period of T seconds. We anticipate that T can be chosen to be from 30-60 seconds, but optimum value will be confirmed after experimental results.

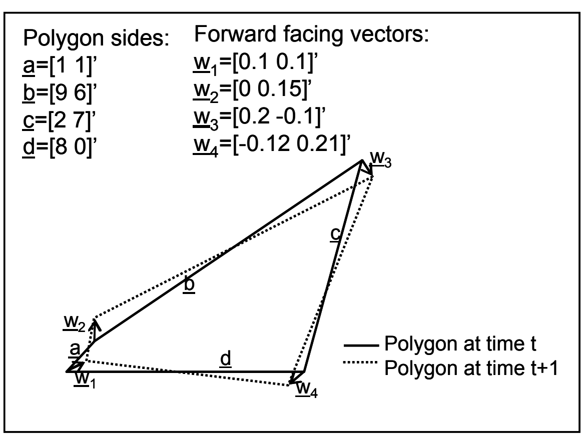

Figure 6: Numerical Example of Simulated Polygons

The graphical representation of Figure 5 exaggerates the magnitude of the forward facing vector for illustration purpose. In the following, we describe a MathLab simulation result that confirms the claim regarding the behavior of the error function. In this simulation, we start with a 4-node polygon at time *t*. From time *t* to *t* +1, the nodes move for a vector displacement given by w_i , i = 1..4, for nodes *a, b, c, d,* respectively. The w_i's are the forward facing vectors as described above.

In the MatLab simulation, we rotate the original polygon of Figure 6 in the clockwise at an increment of 5^{o}. Each rotation represents a clique orientation error of 5^{o}. For each rotated position, we perform the search part of the PMA and find the best reference polygon that matches the forward facing vector direction and the smallest error compared to the UWB polygon at t +1. Thus the PMA will be fully simulated for each rotated orientation. We will record the computed error function given by:

$$P(\theta) = \sum_{t=t0}^{t0+1} \sum_i | R_i - U_i | \quad (4)$$

Notice equation (2) is very similar to equation (1) except that T=1, and that θ cover all the possible orientations, in 5^{o} increments. The result of the simulation is shown in Figure 7.

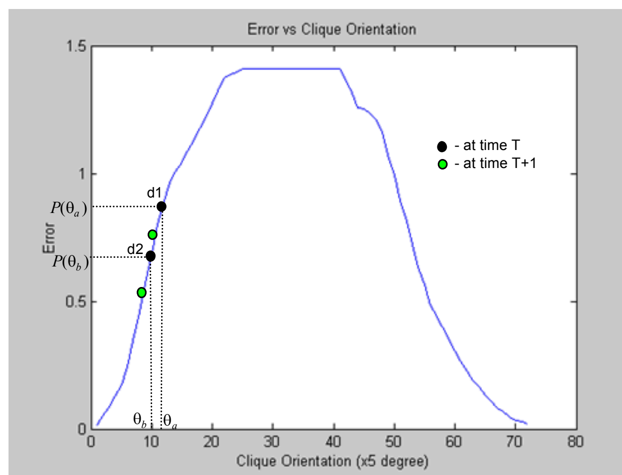

Figure 7: Simulation Showing Polygon Error as a Function of Clique Orientation Deviation

From Figure 7, we notice that the error function assumes a larger value when the clique is rotated away further from the correct orientation in a monotonic manner, although it is not symmetrical with respect to direction of the rotation. This means that the correct orientation corresponds to the minimum of the error curve, which is the most important property to be used in the OPMA. Consider, at any point in time, the clique orientation deviates from the correct orientation by an angle of say 50 °, as shown as “d1” point in Figure 7, we would want to guide the clique so that it moves towards the correct orientation. To steer the system in the right direction, we need to know which direction should the clique be rotating, and by how much. Moreover, this steering should be adaptive, allowing dynamic orientation adjustment all the time. The OPMA operates in the following way. At a given time, each node starts with two orientations θ_a and θ_b, such that $\theta_a = \theta_b + q$, where q is a design parameter. As an example, q is set to 10°. For each update instance T, the error function values $P(\theta_a)$ and $P(\theta_b)$ are computed and compared. Depending on which value is larger, θ_a and θ_b for the next update period is adjusted according to the following rule:

$$\text{if } P(\theta_a) > P(\theta_b) \text{ at T} \quad \theta_a(T+1) = \theta_a(T) - \rho, \quad \theta_b(T+1) = \theta_b(T) - \rho \qquad (5)$$

$$\text{if } P(\theta_a) < P(\theta_b) \text{ at T} \quad \theta_a(T+1) = \theta_a(T) + \rho, \quad \theta_b(T+1) = \theta_b(T) + \rho$$

where ρ is a design parameter. As an example ρ may be set to 5°.

The simulation result suggests that the clique orientation can be adjusted according to two measurements, namely, $P(\theta_a)$ and $P(\theta_b)$. Figure 8 illustrates a flow chart of OPMA adaptation.

V. SMART RELAY DESIGN

The SATS smart relay is designed to be simple and efficient. The SATS relay system is smart in that a lead SATS node will detect when it is wandering too far from the command node and will signal the leader (marine) via a beeping sound and a pop-up menu for deploying a relay. Deploying a relay means that the marine leader will turn on a relay and lay it at some convenient location. Once deployed, the relay will retransmit all packets it receives. The design of the smart relay strives to make it simple and robust.

Consistent with the simple and low-power requirement of smart relay, the SATS relay uses UDP broadcast packets. Here is how it works: The command node sends out Hello packet at the rate of 20 Hello’s per second. All the SATS nodes will re-broadcast all received packets once. The node will recognize packets via the sequence number and packet type field (see Figure 15) and discard those that arrive the second time. One special SATS node is designated the lead node which is responsible for detecting its range from the command node. The lead node will monitor the number of Hello packets received. When the number of Hello’s falls below a configured percentage, say p1, a beeping sound and a pop-up screen will be triggered at the lead node. The leader then deploys relay #1 by turning on the power, and placing it at a convenient location. The clique can then continue on its expedition. Once deployed, the relay will re-broadcast all the packets it receives once. Again, when the same packet arrives at the relay the second and subsequent times, it will be discarded.

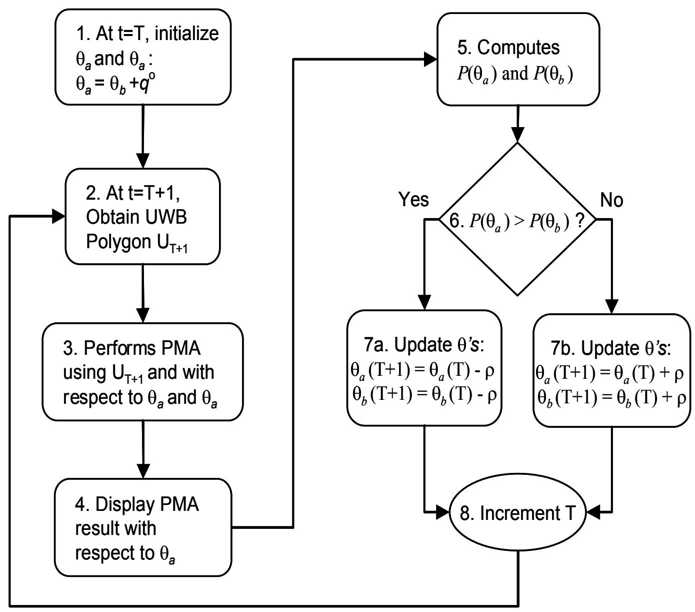

Figure 8: Flow Chart Showing the Adaptive Orientation-correcting PMA

When the lead node wanders far enough from the first relay, the whole process of detection of hello packet threshold, relay deployment notification, relay deployment, is repeated. For the second time, the threshold is set to p2. Exactly the same procedure applies for the third relay deployment with threshold set to p3.

The setting of the threshold p1,p2,p3 is crucial to the performance of relays. For the SATS relays, we found by experience that the following thresholds work well: p1=80%, p2=50%, p3=30%. Since 20 Hello’s are sent by the command station per second, the threshold of the first relay deployment is 16 Hello’s, the second relay threshold is 10 Hello’s, and the third relay threshold is 6 Hello’s.

The smart relay is implemented using Gumstix microcomputer. It is very light-weight, small size, has no display, and uses one Li-ion battery for power. The Gumstix has a mother board (computer) and a WiFi card and uses a Linux operating system. Software that runs on the Gumstix is developed on a Laptop and executable files are downloaded into the Gumstix via a USB port.

VI. RESULTS

The SATS was thoroughly tested for extended operations (2 hours). The highlight the qualitative aspects of the evaluation are given below:

- All functions listed in section 2 were demonstrated to be working properly.
- The Relative Position Error (RPE) was found to be always within 5m.
- With the respect to the Clique Drift Error (CDE), it was in the 60m range.
- The smart relays were operational and properly indicated to the lead node when relays should be deployed. Using 3 relays, an extended distance of 150-200m between the command station and the lead node of the clique was achieved.

From the testing, we learned a few things that can be improved,

- The SATS design did not take into account that the SATS nodes spend a lot of time being stationary and the algorithm did not take that into account and subsequently exaggerate the random movement when the nodes are stationary. Providing a detection of motionless state would potentially reduce the random drifting error.
- Currently 3 smart relays are provided and the smart relay algorithm is not extensible with respect to adding more relays. If more relays are to be desirable, a more reliable protocol can be designed to support any number of relays.

VII. SUMMARY

This paper describes the Self-Correcting Adaptive Tracking System (SATS) algorithm and its implementation. The core of the SATS algorithm performs a search and optimization procedure to find the best new locations subject to constraints provided by updated ranging information and per-node magnetometer information. In addition, SATS has an orientation error tracking feedback loop to perform self-correction of clique level orientation errors.

Test and evaluation showed that SATS performed all the required functions including tracking of local members of the clique, relaying location information back to the command node via 3 smart relays. Display of forward facing view of north facing view, automatic detection of SATS node being carried at the waist level or being held and read position, text broadcast capability and detection and alarm if foreign WiFi is detected.

The SATS performed satisfactory with respect to local relative positions among clique members, but absolute position error is larger than expected (about 60m) after 2 hours of operations. Nonetheless, this is one of the first systems demonstrating operations for 2 hours and still produces reasonable results.

The smart relay algorithm and implementation was found to be satisfactory and allows an extension of 200m between the command station and the clique lead node. Further improvement of the smart relay can allow deployment of more relays and more flexibility with respect to relay deployment procedure.

VIII. ACKNOWLEDGEMENT

The authors would like to acknowledge Office of Navy Research program officers John Moniz and Nathan Smith for their vision and support. We also want to thank Lester Foster of MultiSpectral Solutions, Inc., for his support on the Ultra WideBand technology.

IX. REFERENCES

[1] ONR BAA 06-007, "Navigation in a GPS-denied Environment," http://www. onr.navy.mil/ 02/BAA/docs/baa_06_007.pdf

[2] Eric Foxlin, "Pedestrian Tracking with Shoe-Mounted Inertial Sensors," IEEE Computer Graphics and Applications, November 2005

[3] J. A. Farrell & M. Barth (1999). *The Global Positioning System & Inertial Navigation.* McGraw-Hill. New York.

[4] M. S. Grewal, L. R. Weill, and A. P. Andrews (2001). *Global Positioning Systems, Inertial Navigation, and Integration.* John Wiley & Sons. New York.

[5] Robert J. Fontana. *et. al.* Commercialization of an Ultra Wideband Precision Asset Location System. Proc IEEE Conf. on UWB Systems and Technologies, November, 2003, Reston, VA.