Decentralised SLAM with Low-Bandwidth Communication for Teams of Vehicles

Eric Nettleton[a], Sebastian Thrun[b], Hugh Durrant-Whyte[a] and Salah Sukkarieh[a]
[a]Australian Centre for Field Robotics, University of Sydney, NSW 2006, Australia
[b]Carnegie Mellon University, Pittsburgh, PA, 15213, USA
Email: {ericn,hugh,salah}@acfr.usyd.edu.au

***Abstract*— This paper addresses the problem of simultaneous localization and mapping (SLAM) for teams of collaborating vehicles where the communication bandwidth is limited. We present a novel SLAM algorithm that enables multiple vehicles to acquire a joint map, but which can cope with arbitrary latency and bandwidth limitations such as typically found in airborne vehicle applications. The key idea is to represent maps in information form (negative log-likelihood), and to selectively communicate subsets of the information tailored to the available communication resources. We show that our communication scheme preserves the consistency, which has important ramifications for data association problems. We also provide experimental results that illustrate the effectiveness of our approach in comparison with previous techniques.**

I. Introduction

The problem of simultaneous localisation and mapping (SLAM) has received significant attention in the past few years. While almost all research is focused on the single vehicle case, the multi-vehicle case has almost entirely been overlooked. This is at odds with the fact that many of the most promising robotics domains (military and civilian) involve teams of robots that jointly acquire maps [9], [11], [1]. Decentralised SLAM addresses the problem in which large numbers of vehicles cooperatively acquire a joint map, while simultaneously localising themselves relative to the map. For a decentralised architecture to be truly scalable, the individual communication requirements must be independent of the number of vehicles in the system. For it to be robust, it must be able to accommodate arbitrary latencies and bandwidth limitations in the communication network.

Most conventional SLAM algorithms are based on a seminal paper by Smith and Cheeseman [10]. This paper introduced the Extended Kalman filter (EKF) solution to the SLAM problem, which has been the basis of hundreds of contemporary implementations [5]. The EKF represents maps by the mean and covariance of a Gaussian distribution. Unfortunately, the communication of these entire maps for multi-vehicle problems is not scalable for large map sizes, as it requires the transmission of all N^2 elements in the covariance matrix to remain consistent.

This paper presents a new algorithm for multi-vehicle SLAM. Just as in our previous work [8], we use the information form of the EKF to represent information acquired by the vehicle. The update of the information form is additive; as a result, incremental new information can be integrated across different vehicles with arbitrary network latencies. However, our previous work required communication bandwidths *quadratic* in the size of the map. This paper proposes an efficient communication scheme which makes it possible to communicate updates whose size is independent of the size of the map. By doing so, our approach can cope with arbitrary bandwidth limitations in multi-vehicle SLAM. Our approach is fully decentralised and can (in principle) scale to any number of vehicles.

The ability to communicate maps of arbitrary size is obtained by implementing a hybrid information filter/Covariance Intersect (CI) [4] algorithm on each communication link. This algorithm is implemented separately from the SLAM filter and is used solely to manage the inter-vehicle communication and ensure that information vehicles have shared does not get 'double counted'. The formulation of the problem in information space is also exploited by using information metrics to maximise the amount of information in any communicated submap. Our approach is to communicate those features with the greatest information gain that has not yet been sent. The simulated results included in this paper illustrate that this algorithm functions both consistently and accurately.

The work described in this paper is part of the Autonomous Navigation and Sensing Experimental Research (ANSER) project and is aimed at the development of a multiple flight vehicle demonstration of decentralised SLAM. The system under development consists of four unmanned flight vehicles, each equipped with GPS and inertial sensors and two terrain payloads; a vision system and either a mm-wave radar or laser sensor. However, to implement decentralised SLAM on this system it is necessary to have an algorithm which can operate in a bandwidth constrained environment. The approach presented here overcomes this important obstacle and provides a technique

that is scalable.

II. The Decentralised Architecture

The decentralised architecture used in this research is based on the information or inverse covariance form of the Kalman filter, and builds on the work of Grime [2] on decentralised tracking. It works by propagating *'information'* to all vehicles or nodes in the network. The information is communicated using point-to-point links with no loops in the network, as illustrated in Figure 1. The internal structure of each of these sensing nodes is illustrated in Figure 2.

This section gives an overview of the key elements of the architecture. A more detailed description can be found in [7].

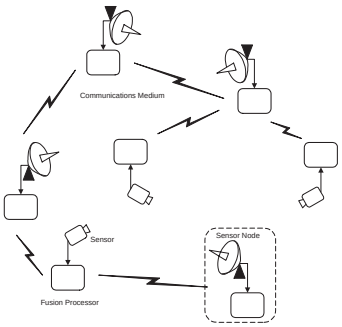

Fig. 1. The general structure of a decentralised data fusion system employing point-to-point communications.

A. The Information Filter

Using the conventional EKF representation of SLAM, the augmented state vector of the vehicle and map at time t_i given observations up to time t_j is written as $\hat{\mathbf{x}}(i \mid j)$ with an associated covariance $\mathbf{P}(i \mid j)$. However, when written in its equivalent information form, the system is given by the information vector $\hat{\mathbf{y}}(i \mid j)$ and information matrix $\mathbf{Y}(i \mid j)$.

$$\hat{\mathbf{y}}(i \mid j) \triangleq \mathbf{P}^{-1}(i \mid j)\hat{\mathbf{x}}(i \mid j), \qquad \mathbf{Y}(i \mid j) \triangleq \mathbf{P}^{-1}(i \mid j).$$

A complete derivation of the non-linear information filter, its propagation and update equations and its application to SLAM is contained in Thrun et al [12]. Maybeck [6] also includes a derivation of the information filter from the standard Kalman filter equations.

The interesting fact is that with regards to information integration, all update operations are *additive*. Addition is communicative. As a result, the specific order in which updates from other vehicles are applied is irrelevant to the results, provided the features do not change over time. This observation is of central importance in multi-vehicle SLAM, as the map features are specifically selected to be stationary. If map update messages from other vehicles arrive under arbitrary latency, they can still simply be added on in the information form regardless of their "age."

B. Local Filter

The local filter is the local implementation of the SLAM algorithm, which generates information state estimates on the basis of observed, predicted and communicated information. Other infrastructure such as the channel filter and channel manager exist only to support the communication of information to and from other vehicles.

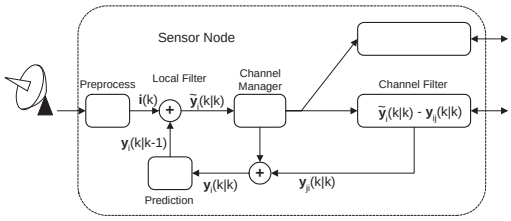

Fig. 2. Structure of a decentralised node.

C. Channel Manager

The channel manager serves as the interface between the local filter and the channel filters (and through these, the other vehicles in the network). The channel manager collects incoming data from the channels at the time horizon, assimilates it using the additive update equations, and then communicates the result to the local filter which can update the SLAM estimate in a single step. It also receives outgoing updated information states from the local filter and disseminates this to the channel filters for transmission.

D. Channel Filter

The channel filter is used to manage communication of map information between nodes, and as such is the focus of the algorithm presented in this paper. It serves two main functions; to keep track of information previously communicated on the channel (to ensure data is not double counted), and to synchronize incoming and outgoing information with the local SLAM filter. Information previously communicated is used to compute new information gain from other vehicles in the network. Synchronization serves to accommodate any delays in information or differences in timing between node filters at remote sites.

Information arrives asynchronously at each channel from remote vehicles. The channel filter calculates the new information received on any given channel and transmits this to the channel manager, before updating itself. In the event that the channel becomes blocked or disconnected, the channel filter effectively fuses the new data and cycles to the next available communication time.

The following section discusses the channel filter algorithm and how it can be formulated to handle the communication of submaps of an arbitrary size.

III. Constant Time Communication

In SLAM, updates come in two forms: measurement and motion updates. While each measurement update affects only a small number of values in the information form, motion updates modify *all* such values. Thus, after each motion command, previous techniques communicate all values in the filter, of which there are quadratically many in the size of the map. Such an approach imposes serious scaling limitations on multi-vehicle SLAM.

Our approach communicates arbitrarily small maps by carving out submaps of maximum information value. The idea is that the vehicle operates in the vicinity of specific landmarks in the map; hence communicating the submap that corresponds to those landmarks yields small messages whose size is independent of the total map size.

Unfortunately, such a scheme will inevitably lead to over-confident estimates if implemented using a standard information or Kalman filter: this is a side effect of the approximation of sending submatrices of the full update (which affects all parameters in the filter). Our new approach composes such sub-matrices in ways that maintain *map consistency*, that is, they guarantee that no vehicle ever overestimates its confidence in individual feature locations. This result is obtained by implementing a hybrid information filter/covariance intersect (CI) [4] algorithm in the communication channels. As the vehicles are communicating their information state estimates of selected map features, they will have information in common which correlates them. In order to cope with this issue, an information estimate is maintained in the communication channel which keeps a record of this 'common information' separately from the SLAM filter. When information submaps arrive from another vehicle, they are fused using CI with this common information to yield a consistent, but conservative update. Since the update of an information filter is additive, the effective increment of new information that the communicated submap contributed is given by the difference between the channel estimate before and after the CI update. This increment of map information can then be safely added to the SLAM filter, completing the update cycle. It is important to note that the use of the conservative CI update is limited strictly to the communication algorithm, while the SLAM estimator is implemented using the information form of the Kalman filter.

Another key aspect of this algorithm is that the submaps being communicated are not static submaps such as those used to aid computational efficiency, but instead are a dynamically selected set of features chosen at each communication step based on some heuristic. It is this ability to formulate submaps of arbitrary size that allows the communication strategy to operate in constant time. Also, by formulating the problem in its information form, it is a simple matter to maximise the amount of information being transmitted in any given submap by selecting features which have experienced the greatest information gain.

The ability to communicate constant time maps is also achievable using other approaches such as that proposed by Williams [13]. However, this approach is not robust to communications failure as it essentially transmits increments of map data. If any increment is lost, it is irrecoverable. When formulated in the information form presented here it is possible to transmit the actual state estimate that contains all prior information about the map features. Using this approach, each communication automatically contains all information in prior messages as well as any new data. In order not to double count data, the channel filter is used at reception to subtract the common information and pass only the increment of new information to the SLAM filter.

The Split Covariance Intersect (SCI) algorithm proposed by Julier [3] is similar to the approach presented here. In uses a hybrid CI/Kalman filter approach to manage the complexity of large maps, and in the process notes that the same approach can be extended to multiple vehicles. However, as the SCI method does not use the channel filter concept to track any information nodes have in common, it is not able to make total use of information should complete maps ever be transmitted.

A. Extracting the Submap to Communicate

Our approach to selecting features to communicate is to dynamically select those which *maximise the information gain* down any particular channel. This is done using the features that have the greatest amount of information that has not yet been communicated to other nodes. As the problem is already formulated in information space, this is a trivial exercise and can be done simply by subtracting the amount of information transmitted (recorded in the channel filter) from the current information state. A submap of some arbitrary size M^2, where $0 \leq M \leq N$, can then formed using the M features with the *greatest information gain*. This ability to form an arbitrary size submatrix to communicate is a powerful concept as it allows the algorithm to function without needing to communicate the entire map of size N^2. While it is possible to use practically any method of selecting the features to send, the 'maximum information gain' technique is used in this paper with highly satisfactory results.

Once the features to transmit have been selected, it is necessary to integrate out the effect of all of the other states from this submap. This can be done by defining a projection matrix G_m such that $G_m\hat{\mathbf{y}}(k \mid k)$ extracts only those features in the map which are to be transmitted,

and another projection G_v such that $G_v\hat{\mathbf{y}}(k \mid k)$ contains all other states. Using the notation $\mathbf{y}^*(k \mid k)$ and $\mathbf{Y}^*(k \mid k)$ to define the information vector and information matrix of the submap to be sent, we integrate out the effect of the other states:

$$\begin{aligned}\mathbf{y}^*(k \mid k) &= G_m[\hat{\mathbf{y}}(k \mid k) - \mathbf{Y}(k \mid k)G_v^T \\ &\quad \times \left(G_v\mathbf{Y}(k \mid k)G_v^T\right)^{-1} G_v\hat{\mathbf{y}}(k \mid k)] \\ \mathbf{Y}^*(k \mid k) &= G_m[\mathbf{Y}(k \mid k) - \mathbf{Y}(k \mid k)G_v^T \\ &\quad \times \left(G_v\mathbf{Y}(k \mid k)G_v^T\right)^{-1} G_v\mathbf{Y}(k \mid k)]G_m^T(1)\end{aligned}$$

The small information submap given by $\mathbf{y}^*(k \mid k)$ and $\mathbf{Y}^*(k \mid k)$ is now sent to the channel filters prior to transmission to another node.

B. Channel Update

As the channel filter maintains an estimate of the common information between nodes, the only states it will have are those that have been communicated. Since the vehicle information is never communicated, the channel filters will never maintain any states other than the map. Therefore, as map features are by definition stationary, the prediction model of the channel filter is an identity matrix and the state at time t_{k+1} is equal to the state at t_k.

If the full map of N^2 features is being transmitted, the channel filter can be updated simply using:

$$\begin{aligned}\mathbf{Y}_{Chan}(k \mid k) &= \mathbf{Y}_{Chan}(k \mid k-1) + \\ &\quad [\mathbf{Y}^*(k \mid k) - \mathbf{Y}_{Chan}(k \mid k-1)] \\ &= \mathbf{Y}^*(k \mid k) \\ \hat{\mathbf{y}}_{Chan}(k \mid k) &= \hat{\mathbf{y}}_{Chan}(k \mid k-1) + \\ &\quad [\mathbf{y}^*(k \mid k) - \hat{\mathbf{y}}_{Chan}(k \mid k-1)] \\ &= \mathbf{y}^*(k \mid k) \qquad (2)\end{aligned}$$

This simple update is only possible as the map information $\mathbf{Y}^*(k \mid k)$ and $\mathbf{y}^*(k \mid k)$ are of the dimension of the complete map and all cross information terms are being transmitted. For the constant time communication algorithm this is not the case and a different update must be performed.

The channel filter update when sending submaps of arbitrary size is performed using CI. While this method will yield a conservative result, it is necessary in the absence of the complete N^2 map information estimate. When the information submap given by $\mathbf{y}^*(k \mid k)$ and $\mathbf{Y}^*(k \mid k)$ arrives the CI update of the channel filter can be done by inflating the submap to the state dimension using an appropriate projection G_m. Selection of the CI weighting parameter ω is done to minimise the determinate of the result.

$$\begin{aligned}\mathbf{Y}_{Chan}(k \mid k) &= \omega\mathbf{Y}_{Chan}(k \mid k-1) + \\ &\quad (1-\omega)\, G_m^T\mathbf{Y}^*(k \mid k)G_m \\ \hat{\mathbf{y}}_{Chan}(k \mid k) &= \omega\hat{\mathbf{y}}_{Chan}(k \mid k-1) + \\ &\quad (1-\omega)\, G_m^T\mathbf{y}^*(k \mid k) \qquad (3)\end{aligned}$$

Following this process the channel filter is completely updated and the information submap given by $\mathbf{y}^*(k \mid k)$ and $\mathbf{Y}^*(k \mid k)$ is transmitted to other nodes. It is important to note that this submap contains the entire state history of those features in it and not simply an increment of information or a batch of observations. In formulating the problem this way, the algorithm is robust to any communication failures as every message contains not only new information, but all information in previous messages as well. As the information from any previous messages is stored in the channel filter as common information, it can be removed easily by the receiving node to ensure that information is not fused more than once. For example, if a particular submap of features is transmitted but for some reason never received, the information in the lost message is automatically recovered next time those features are communicated.

When the receiving node gets the new submap information, it updates its own channel filter using exactly the same update steps as above. Once updated, it calculates the increment of new information it has just received from node i that has not already been fused locally at node j.

$$\begin{aligned}\mathbf{Y}_{ij}(k \mid k) &= \mathbf{Y}_{Chan}(k \mid k) - \mathbf{Y}_{Chan}(k \mid k-1) \\ \hat{\mathbf{y}}_{ij}(k \mid k) &= \hat{\mathbf{y}}_{Chan}(k \mid k) - \hat{\mathbf{y}}_{Chan}(k \mid k-1) \qquad (4)\end{aligned}$$

This information increment is then sent to the local filter to be fused into the SLAM estimate.

C. Fusing Information from Other Nodes at the Local Filter

When the local filter receives the information $\mathbf{Y}_{ij}(k \mid k)$ and $\hat{\mathbf{y}}_{ij}(k \mid k)$ from the channel filter it must use this information in the SLAM estimate. In order to do this, it is necessary to firstly define a matrix G_s that inflates the map to the dimension of the entire SLAM state by padding vehicle elements with zeros.

The update is then done by adding the new information from the other node as:

$$\begin{aligned}\hat{\mathbf{y}}(k \mid k) &= \hat{\mathbf{y}}(k \mid k-1) + G_s\hat{\mathbf{y}}_{ij}(k \mid k) \\ \mathbf{Y}(k \mid k) &= \mathbf{Y}(k \mid k-1) + G_s\mathbf{Y}_{ij}(k \mid k)G_s^T \qquad (5)\end{aligned}$$

It is worth noting that this update step is identical to updating with information from locally attached sensors.

IV. Results

The constant time communications algorithm was implemented and run in a multi-vehicle simulation to test and evaluate its performance with respect to a number of other communication strategies. The different communications strategies used were:

1) No communication between platforms - each vehicle operates independently using only its own observations.

2) Transmission of the complete information map of N^2 elements at every communication step.
3) The constant time communications strategy where a submap of only 5 features were transmitted at each communication step. The features to include in the transmitted submap were selected at transmission time to be those which had the greatest amount of information not yet sent.

The simulation used 100 vehicles moving in an environment of 100 features. Figure 3 (h) illustrates this simulation world, marking the feature locations and a number of the vehicle paths. A non-linear tricycle motion model was implemented to estimate the position and orientation $\begin{bmatrix} x & y & \phi \end{bmatrix}$ of each vehicle within the map. Each was equipped with a range/bearing sensor which gave observations to features in the forward hemisphere of the vehicle at a frequency of 1Hz. Vehicles were able to communicate map information to their neighbours in the network at a frequency of 0.1Hz.

The results of the simulation are illustrated in Figure 3. The benefit of communicating information can be seen in Figure 3 (a)-(g) as the error and 2σ bounds of both the N^2 and constant time communication algorithms are significantly better than the case when the vehicles do not communicate at all. Of the communication schemes, the N^2 performs the best as it always propagates the entire map. However, the constant time communications algorithm can be seen to give results that are only slightly worse than this. Furthermore, it does this by transmitting submaps of at most 5 features each communication step, instead of all 100. Since the number of elements required for communication is quadratic in the size of the map, the bandwidth savings for this problem are significant when using the constant time method.

The results indicate that the constant time algorithm approaches the N^2 strategy quite quickly even though it is not transmitting as much information. This occurs as the submap that is transmitted is dynamically selected to contain those features which will give the greatest information gain. This trend illustrates the idea that good information about a small number of features increases the map accuracy significantly. This concept is further illustrated in Figure 3 (i) which shows that although increasing the size of the communicated submaps does increase the quality of the estimates, the rate at which the quality changes decreases as the submaps get larger. This is to be expected as communication of the whole map will include features that have not been observed recently and which will therefore not contribute large amounts of new information. Communicating larger submaps using this constant time algorithm simply has the effect allowing the estimate to converge toward the N^2 result faster.

V. Conclusion

This paper has presented an algorithm for constant time communications in multiple vehicle SLAM problems. Using this method, the communication requirements for the decentralised system are independent of the number of features in the SLAM map. It was shown that the effect of communicating these smaller submaps maintains map consistency while providing only slightly less information than transmitting the entire map. However, should there be an opportunity to transmit the complete map of N^2 features, the algorithm is able to make complete use of the information and recover any information discarded through previous conservative CI updates.

ACKNOWLEDGMENTS

The first author is funded by BAE SYSTEMS Australia and would like to thank them for his PhD scholarship.

References

[1] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2000.

[2] S. Grime and H.F. Durrant-Whyte. Data fusion in decentralized sensor networks. *Control Engineering Practice*, 2, No 5:849–863, 1994.

[3] S. Julier and J. Uhlmann. Building a million beacon map. In *Sensor Fusion and Decentralised Control in Robotic Stystems IV*, volume 4571, pages 10–21, 2001.

[4] S. Julier and J. Uhlmann. General decentralised data fusion with covariance intersection (ci). In D. Hall and J. Llinas, editors, *Handbook of Data Fusion*. CRC Press, 2001.

[5] J. Leonard, J.D. Tardós, S. Thrun, and H. Choset, editors. *Workshop Notes of the ICRA Workshop on Concurrent Mapping and Localization for Autonomous Mobile Robots (W4)*. ICRA Conference, Washington, DC, 2002.

[6] P.S. Maybeck. *Stochastic Models, Estimation and Control, Volume 1*. Academic Press Inc., New York, 1979.

[7] E. Nettleton, H. Durrant-Whyte, and S. Sukkarieh. A robust architecture for decentralised data fusion. In *Proceedings of IEEE International Conference on Advanced Robotics*, 2003.

[8] E.W. Nettleton, P.W. Gibbens, and H.F. Durrant-Whyte. Closed form solutions to the multiple platform simultaneous localisation and map building (SLAM) problem. In *Proc. SPIE*, volume 4051, pages 428–437, 2000.

[9] R. Simmons, D. Apfelbaum, W. Burgard, M. Fox, D. an Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. In *Proceedings of the AAAI National Conference on Artificial Intelligence*. AAAI, 2000.

[10] R.C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4):56–68, 1986.

[11] C. Thorpe and H. Durrant-Whyte. Field robots. In *Proceedings of the 10th International Symposium of Robotics Research (ISRR'01)*, 2001.

[12] S. Thrun, D. Koller, Z. Ghahmarani, and H. Durrant-Whyte. SLAM updates require constant time. In *Proceedings of the Fifth International Workshop on Algorithmic Foundations of Robotics*, Nice, France, 2002.

[13] S. Williams. *Efficient Solutions to Autonomous Mapping and Navigation Problems*. PhD Thesis, University of Sydney, 2001.

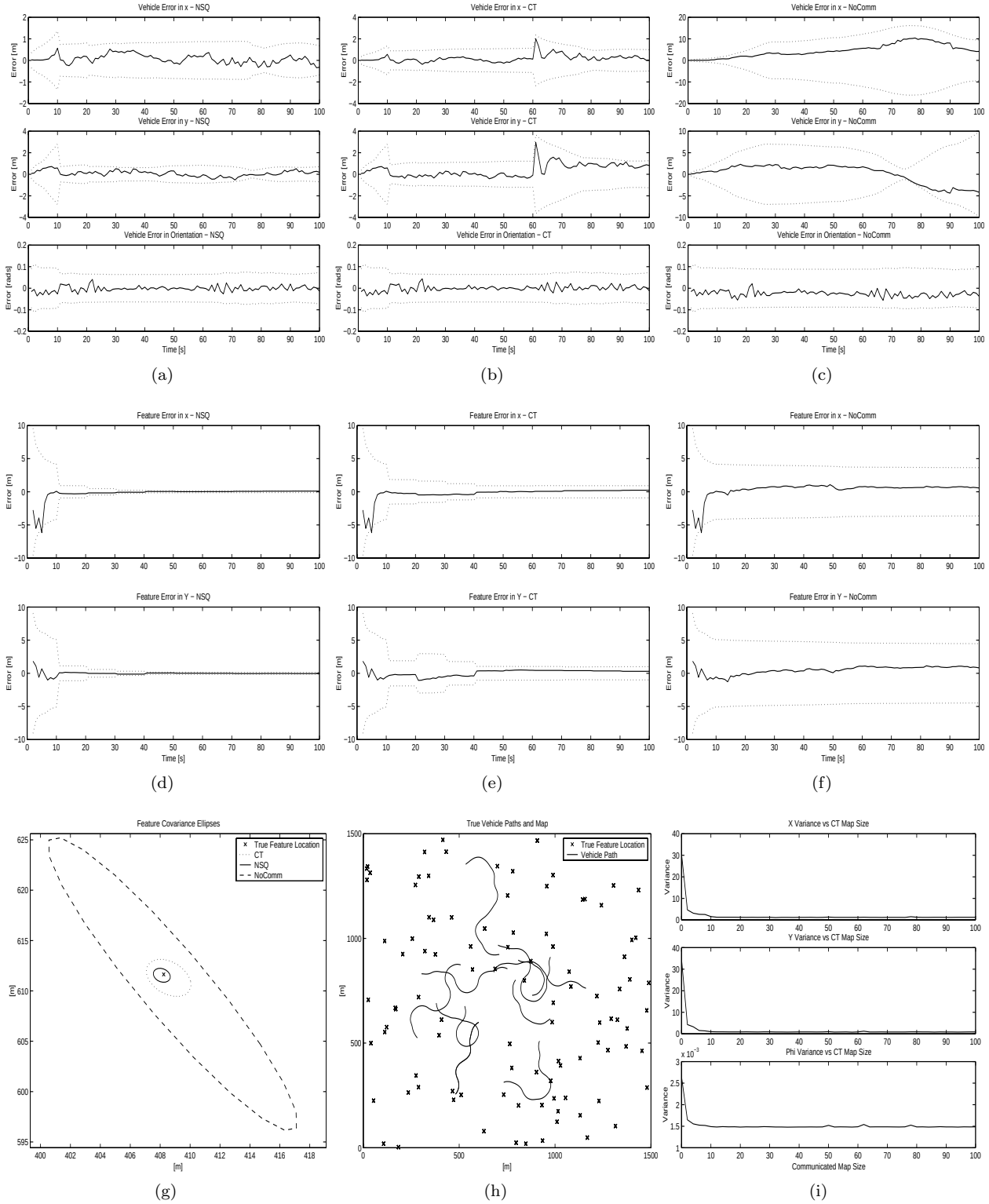

Fig. 3. Simulation results. The error and 2σ bounds for the vehicle states are given for the (a) N^2, (b) constant time and (c) no communications strategies. The error and 2σ bounds for a typical feature are also given for the (d) N^2, (e) constant time and (f) no communications strategies. The covariance ellipse for a feature under the different communication strategies is illustrated in (g). The simulation world is shown in (h). For clarity, only 10 of the 100 vehicle paths are shown. (i) shows the mean variance in x, y and ϕ from all vehicles over the period of the simulation for different maximum map sizes using the constant time communications algorithm.