**KimonoNet  
Enabling Efficient and Reliable Inter-UAV   
Fluid Data Communication Link**

*Document*  
Version 1.0.01  
March 20, 2012

*Authors*  
Bollens, Eric <ebollens@oit.ucla.edu>   
Hung, James <james.h@ucla.edu>  
Khalapyan, Zorayr <zkhalapyan@oit.ucla.edu> Norris, Wade <wade.norris@ucla.edu>

*KimonoNet addresses a peer-to-peer network research topic related to routing and transport control issues in sparse networks of highly mobile ad hoc peers such as unmanned aerial vehicles. Traditional distance-vector and link-state algorithms are not suited for such a topology given the high mobility and churn of its constituents, which causes routes to shift too quickly for global route propagation. Therefore, this approach leverages a coordinate-based routing system proposed by Greedy Perimeter Stateless Routing and adds two-hop neighbor awareness to minimize control packets and improve reliability in fluid, sparse network graphs.*

# 1. Introduction

## 1.1. Background Information

In recent years, the military has increased computing power in the field significantly; however, the increase in network coverage has not followed suit. One option to increase coverage is deployment of an ad hoc network whereby network-enabled equipment may form routes through a remote operational zone back to a secure base station.

Rather than providing the route via stationary relays, which make for easy targets, unmanned aerial vehicles and other mobile equipment with basic network capabilities may provide such a network in hostile environments. However, a network manifest as a very sparse graph with limited routes to any endpoint. Further, given the mobility of nodes and possible calamities that might befall them, the network may also suffer high churn. Consequently, an implementation must adapt quickly to topological changes and provide reasonable service even when individual route exist only briefly.

Beyond military application, this research topic may provide insight into broader issues in ad hoc networking. To address the dynamics of ad hoc networks, a number of protocols including Ad hoc On-Demand Distance Vector Routing (AODV) and Dynamic Source Routing (DSR) have taken the approach of on-demand routing, ascertaining a path at send-time rather than predetermining routes. However, these protocols suffer from disadvantages including long setup times, inconsistency and unreliability. Other ad hoc protocols such as the Optimized Link State Routing Protocol (OSLR) employ a table-driven approach reminiscent to traditional distance-vector protocols, but they struggle to adequately handle the rapidly changing nature of such a network with inordinate control traffic.

## 1.2. Objectives

Seeking to minimize both setup time and control traffic, this project foregoes traditional distance-vector and link-state protocols, as well as on-demand approaches. Instead, it uses Greedy Perimeter Stateless Routing, a coordinate-based approach first proposed by Karp and Kung [], and then extends it by propagating two-hop neighbor knowledge through control beacons. This strategy increases the interval between beacons, thus reducing control traffic; it also provides the foundation for other improvements based on this knowledge.

To these ends, this project provides a comprehensive description of the KimonoNet protocol including algorithms, message formats, structures, states and functions. While this document summarizes its key principles, a full description of the protocol may be found in *KimonoNet: Protocol Specification Document* [].

Further, this project provides a KimonoNet client prototype that runs under a JVM and provides both a production mode for field deployment and a test mode for simulations.

Finally, this project includes results from simulation that measure the viability and effectiveness of KimonoNet through metrics including packet delivery, routing overhead, average latency and path optimality.

## 1.3. Scope

### 1.3.1. Network Layers

The KimonoNet protocol is an ad hoc mesh overlay. Optimally, this protocol should be implemented directly over Layer 2. However, its flexibility also supports implementations on Layers 3 and 4 that leverage wireless broadcast or multicast.

For convenience, the KimonoNet prototype it built on UDP multicast. To support test mode, it also includes special provisions to simulate multiple nodes in a single environment.

### 1.3.2. Destination Locations

The KimonoNet protocol as described forthwith and in associated material supports mobile source nodes but requires a destination node of known location. In the event that a destination is not static, it assumes that another mechanism will be used to locate the destination. This design decision may present challenges for some communication, but it is purposeful in that it decouples the protocol from search mechanisms, for which significant research has already been devoted.

For simplicity, the prototype client does not provide a search algorithm to locate a mobile destination but instead assumes that the destination is fixed. A flooding search algorithm could be implemented in a future iteration so that a source can locate a mobile destination.

### 1.3.3. Data Payloads

This protocol provides an overlay for routing and transport across a fluid mesh network. As such, the protocol is ambivalent to the data it carries. It relies on the source to package this data and the destination to interpret this data. Consequently, it may transport application-layer data or lower-layer payloads tunneling across KimonoNet.

However, while KimonoNet supports tunneling, the implications of such are not considered forthwith. Users seeking to leverage duplex protocols such as TCP may encounter significant challenges. This is because KimonoNet assumes destinations of known location, and thus requires either a searching algorithm or that the mobile host initiate the communication. However, even when one of these conditions is met, by the time the data propagates to the other party, it may be stale and inaccurate. Large network diameters accentuate this problem.

### 1.3.4. Simplifying Assumptions

This protocol makes two additional simplifying assumptions: (1) it does not consider altitude, and (2) it assumes symmetric communication between any two nodes in the graph whereby, if one node can communicate with another, the reverse is true as well.

## 1.4. Use Cases

Military operations with highly fluid network topologies serve as the motivation for this project. In such scenarios, UAVs and other network-enabled devices move rapidly and may even disappear from the network due to failure or destruction. Consequently, the approach proposed must not require the continuing existence of any node in the network, but it may predict topology changes when possible to help reduce control traffic.

Further, due to the sparse nature of the network graph, the protocol must operate efficiently given limited routing options. The majority of nodes in the network will likely be out of range of any individual node, and thus it must leverage local knowledge and still make near-optimal forwarding decisions.

This routing protocol may extend beyond military operations. It has applicability in any network with high churn where the location and velocity of nodes are generally known. As such, this ad hoc communication protocol could easily support orbiting satellites and maritime expeditions. However, this algorithm will not cover scenarios where location is not available or where velocity changes unexpectedly; in such situations, other approaches are better suited.

## 1.5. Use Cases

Two primary constituents exist in this scenario:

1. Autonomous nodes that send new packets and route received packets.
2. End points that receive data packets routed across the network.

In the motivating scenario, unmanned aerial vehicles meet the former whereas command posts and external uplinks satisfy the latter.

This protocol assumes the initial communication is sent from an autonomous node to an end point of known location. This avoids the need to introduce a search algorithm. However, a full implementation may add such a mechanism to support communication between any pair of nodes in the network and to efficiently support duplex transport.

### 1.5.1. Autonomous Nodes

The primary constituents of the network, autonomous nodes have (1) awareness of their position and velocity, (2) a NIC that supports ad hoc communication, (3) a network layer that supports broadcast, multicast or promiscuous packet delivery, and (4) a running instance of the KimonoNet client. These nodes are regarded as autonomous because they make independent decisions about position and velocity without considering its implications on routing.

Autonomous nodes collect data, accomplish objectives and then seek further instructions. In order to transmit this data or receive further instructions, these nodes introduce data packets into the ad hoc network. These packets are addressed to the known location of an endpoint, and then forwarded through the network based on the routing algorithm.

Beyond introducing packets into the network, autonomous nodes must also receive data packets passed to them by other nodes and then forward them on based on the routing algorithm.

### 1.5.2. End Points

A command post or other external uplink to the Internet serves as the destination endpoint of a packet originating within the ad hoc network. Because this protocol does not consider a mechanism to determine endpoint location, it requires the originator to know the location of the end point and that this position does not change over time.

As with autonomous nodes, end points must also have (1) awareness of their position and velocity, (2) a NIC that supports ad hoc communication, (3) a network layer that supports broadcast, multicast or promiscuous packet delivery, and (4) a running instance of the KimonoNet client.

After receiving a packet from an autonomous node, an end point may use location and velocity information contained within the packet to reply. Because an autonomous node has variable position, all responses from an end point should have QoS classification of time-sensitive and be loss tolerant.

## 1.6. Constraints

### 1.6.1. High Churn

Due to the operating environment, the routing protocol must handle high churn effectively.

Each autonomous node has a velocity that affects its position over time; consequently, the neighborhood for a given node may change swiftly as nodes enter and recede from a network horizon. The routing algorithm handles this by considering two-hop neighbors and extrapolating position over time based on velocity to determine which nodes will be leaving range and which will be entering, thus reducing route maintenance traffic.

Further, given the motivating scenario, nodes may become unavailable unexpectedly due to failure or destruction. The algorithm presented must handle this independent influence on churn adequately, compensating for the fact that a route may become unavailable quickly and without warning. However, this protocol acknowledges that the unexpected disappearance of a node may cause the loss of a packet, but it should take strides to minimize the likelihood.

### 1.6.2. Mixed Horizons

In a sparse network, only a small set of nodes is within range of any individual node; further, adjacent neighborhoods likely share only a few nodes, if any. The routing algorithm must thus take this into account, as adjacent neighborhoods may not be able to communicate directly and a route must instead be established through other nodes to traverse the void.

This constraint is the reason that distance-vector and link-state management is non-optimal: by the time route information has propagated, routes may have changed substantially. This protocol thus operates under the assumption that nodes cannot self-route data through the entire network, and instead that nodes make decisions based solely on their local neighborhood.

## 1.7. Related Work

G. G. Finn proposed greedy route selection for coordinate-based routing in 1987, using a flood-based approach to resolve situations where greedy selection failed []. B. Karp and H. T. Kung reduced routing overhead by providing a method for detouring without flooding by way of graph planarization []. Their work, namely Greedy Perimeter Stateless Routing, serves as the basis for KimonoNet, which is then extended through two-hop awareness for velocity-based prediction of neighborhood changes.

# 2. PROTOCOL

At its heart, the KimonoNet protocol has two primary objectives:

* Nodes provide information that allows other nodes to recognize them as neighbors and predict how their position will change over time.
* Nodes forward data packets through the network to the intended final destination while making near optimal routing decisions at each hop based solely on local network knowledge.

In order to accomplish these goals, KimonoNet leverages work by Karp and Kung on GPSR and extends it such that a node updates its neighborhood proactively based on position and velocity information rather than only when it receives new updates from neighbors.

## 2.1. Beaconing

### 2.1. Beacon Initialization

When a peer first enters the network, it transmits a beacon packet. This beacon packet includes a unique identifier for the node, as well as a location, velocity and timestamp. All nodes within the node’s network horizon receive this beacon and update their routing tables accordingly.

### 2.2. Beacon Updates

At regular intervals, each node transmits a beacon packet similar to the beacon transmitted during initialization. Again, this packet includes a unique identifier for the node, as well as a position, velocity and timestamp. Beyond this information, the beacon includes information about any neighbors that the sender has identified within its network horizon.

Because these beacon packets contain information about both the node and its neighbors, a recipient can build two-hop awareness of both its neighbors and the neighbors of its neighbors, which it may then use to make routing decisions proactively even before it receives updates from these nodes.

### 2.3. Beacon Acknowledgement

When a node receives a beacon that does not include information about itself, it adds the information from the beacon to its own routing table and then responds with a beacon acknowledgement. This response, like recurring beacon updates, should include both the node’s information and information about all nodes within the node’s network horizon.

Beacon acknowledgements exist to reduce the amount of time it takes for a node to learn about its local topology. They provide a form of on-demand awareness whereby, if one node learns about a new node, then this node informs the new node of its existance as well. This procedure will not loop because, once a node learns of another node, it places it in its beacon, and therefore the node will not reply back with its own location again.

## 2.2. Routing

### 2.2.1. Greedy Forwarding

### 2.2.2. Perimeter Forwarding

### 2.2.3. Predictive Neighbor Maintenance

Because beacons contain not only location, but also velocity and time, this protocol can perform simple calculations to predict changes to its neighborhood over time even.

In the case of one-hop awareness, this reduces control packets and increases reliability and efficiency. This is because each node knows can compute how its neighbors’ positions will change over time. A node thus knows more quickly when a neighbor is no longer within its network horizon or when a neighbor has moved to a position that makes it more or less optimal for routing, all without the neighbor having to actually send another beacon.

Further, because a beacon includes information not just about a node but also its neighbors, it is thus possible to have two-hop awareness. This both reduces control packets and increases route efficiency. With a simple computation, a node can determine when a neighbor of a neighbor will enter into its network horizon and thus become a direct neighbor itself. This allows a node to more quickly take advantage of another node in its routing decisions, even before the two have explicitly exchanged beacons.

In order to efficiently implement this scheme, KimonoNet uses two routing tables. From received beacons, it maintains a 2-hop routing table of all nodes learned from the beacons. Asynchronous to other tasks, it then uses this two-hop routing table to compute a one-hop routing table of neighbors directly within its network horizon [ § 3.2]. In the beacon packets that the node sends, it includes information about all nodes in its one-hop routing table.

### 2.2.4. Quality-of-Service

# 3. Implementation

# 4. Simulation

# 5. ConclusionS

## 5.1. Outcomes

## 5.2. Future Work

Provide an additional protocol for finding the location of a mobile node

QoS to include reliability

Hold packets for short duration if greedy node is coming into range

# 6. Acknowledgements

We thank G. Pau, a mentor who advised us throughout this process, which itself was an outcome of his CS 114: Peer-to-Peer Networks class at UCLA.

Further, we thank B. Karp and H. T. Kung for their work on Greedy Perimeter Stateless Routing [], as well as G. G. Finn, who first proposed greedy routing through a coordinate-based system [].

# 7. REFERENCES

[] Bollens, E., Hung, J., Khalapyan, Z., Norris, W. KimonoNet: Protocol Specification Document, University of California, Los Angeles, CA., Mar. 2012.

[] Finn, G. G. Routing and addressing problems in large metropolitan-scale internetworks. Tech. Rep. ISI/RR-87-180, Information Sciences Institute, Mar. 1987.

[] Karp, B. Geographic Routing for Wireless Networks. Ph.D. Dissertation, Harvard University, Cambridge, MA., Oct. 2000.

[] Karp, B., Kung, H. T. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, in Proceedings of the sixth ACM/IEEE International Conference on Mobile Computing and Networking (MOBICON 2000), Boston, MA, Aug. 2000, pp. 243-254.