Identifying surrounds and engulfs relations in mobile and coordinate-free geosensor networks

ALAN BOTH, RMIT University
MATT DUCKHAM, RMIT University
MICHAEL F. WORBOYS, University of Greenwich

This paper concerns the definition and identification of qualitative spatial relationships for the full and partial enclosure of spatial regions. The paper precisely defines three relationships between regions, "surrounds," "engulfs," and "envelops," highlighting the correspondence to similar definitions in the literature. An efficient algorithm capable of identifying these qualitative spatial relations in a network of dynamic (mobile) geosensor nodes is developed and tested. The algorithms are wholly decentralized, and operate in-network with no centralized control. The algorithms are also "coordinate-free," able to operate in distributed spatial computing environments where coordinate locations are expensive to capture or otherwise unavailable. Experimental evaluation of the algorithms designed demonstrates the efficiency of the approach. Although the algorithm communication complexity is dominated by an overall worst-case $O(n^2)$ leader election algorithm, the experiments show in practice an average-case complexity approaching linear, $O(n^{1.1})$.

CCS Concepts: •**Information systems** → **Geographic information systems;** •**Networks** → **Sensor networks;** •**Software and its engineering** → **Peer-to-peer architectures;** •**Theory of computation** → *Computational geometry;*

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Geosensor networks, qualitative spatial reasoning, moving objects, Voronoi regions, decentralized algorithms

ACM Reference Format:
Alan Both, Matt Duckham, and Michael F. Worboys, 2016. Identifying surrounds and engulfs relations in mobile and coordinate-free geosensor networks. *ACM Trans. Spatial Algorithms Syst.* V, N, Article A (January YYYY), 21 pages.
DOI: 0000001.0000001

1. INTRODUCTION

Geosensor networks ("wireless sensor network[s] that monitor phenomena in geographic space" [Nittel et al. 2004]) are a maturing technology, with a wide range of applications, from environmental monitoring to e-health and smart materials [Nittel 2009; Duckham et al. 2014; Zhong et al. 2015].

Making sense of the large volumes of complex spatiotemporal data from such geosensor networks can be aided by the use of *qualitative* spatial relations. Qualitative spatial relations provide discrete domains, smaller than quantitative alternatives, where the relations correspond to distinctions salient to humans [Galton 2000]. Thus, a body of existing research has focused on tracking the qualitative spatial relations of entities monitored by geosensor networks [Jeong and Duckham 2013; Ercan et al. 2013; Malazi et al. 2013; Avci et al. 2014; Jeong et al. 2014].

In line with previous work, this paper proposes the definition of two related qualitative spatial relations—"surrounds" and "engulfs"—both concerned with the partial

This work is supported by the Australian Research Council, under grant DP120103758.
Authors' addresses: A. Both and M. Duckham, School of Science, RMIT University; M.F. Worboys, Department of Computing and Information Systems, University of Greenwich.

DOI: 0000001.0000001

enclosure of a region by one or more other regions. Surrounds and engulfs can be thought of as intermediate stages between when a region is strictly contained, and when it can be regarded as "free" from the influence of other regions. Correspondingly, these relations are precursors to the identification of salient events, such as "capture" or "escape."

The paper also develops and tests new efficient algorithms for identifying these relationships for regions monitored by a network of dynamic (mobile) geosensor nodes. In common with most research into algorithms for geosensor networks today, our algorithms are *decentralized*—a type of distributed algorithm where no single system element has knowledge of the entire system state [Lynch 1996]. Decentralization ensures algorithms can operate in the network with no centralized control, making them more scalable and resilient to network and node failures.

Unlike many decentralized spatial algorithms today, our algorithms are able to operate with *mobile* geosensors. Our algorithms assume individual nodes are free to move around and explore the geographic space, communicating on an ad hoc basis with nearby nodes they may happen to meet. Node mobility is important in a range of applications, in particular where nodes are attached to mobile objects, such as untethered ocean buoys [Nittel et al. 2007]. Of course, our algorithms can also operate for static nodes, a special case of mobility where the speed of movement is zero.

Further, our algorithms are coordinate-free, relying only on imprecise location information about the spatial neighborhoods of nearby nodes within communication range instead of precise coordinate information. Precise, accurate, and ubiquitous coordinate location for nodes is difficult to acquire, especially in low-cost, low-power geosensor networks and when those nodes are mobile. GPS, for example, places high demands upon a mobile node's limited energy budget, and is unreliable in environments such as dense vegetation, urban canyons, or indoors.

In summary, the three key contributions of this paper are:

— the precise definition of the qualitative spatial relations "surrounds" and "engulfs," as well as a third relation "envelops" required for complex areal objects, based on an existing formal model, the *maptree*;
— two algorithms able to efficiently identify "surrounds" and "engulfs" relationships between both simple and general binary regions monitored by mobile geosensor networks;
— an experimental evaluation of the average case complexity of the algorithms using simulation, demonstrating the efficiency of the approach when deployed in a mobile and coordinate-free geosensor network.

The remainder of this paper begins with a review of the relevant literature connected with relevant qualitative spatial relations, including existing "surrounds"-like relationships. Sections 3 and 4 introduce the formal definitions for our "surrounds" and "engulfs" relationships for simple (Jordan) and general (possibly containing holes) binary regions respectively. The design of two efficient, decentralized algorithms capable of computing "surrounds" and "engulfs" relations between simple and general regions respectively is explored in Section 5. Section 6 evaluates these algorithms experimentally in terms of scalability (communication complexity), with Section 7 concluding the paper and identifying the limitations and areas of future work.

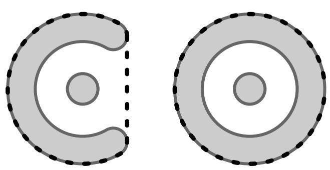
2. RELATED WORK

Region *containment* is a well-established concept. Containment between regions is precisely defined, for example, in the 4-intersection model (4IM) [Egenhofer and Franzosa 1991] (and the later 9-intersection model (9IM) [Egenhofer and Herring 1992]) as a relation between two regions where the boundary and interior of one region exists entirely within the interior of the other.

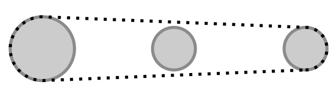
By contrast, the spatial relation "surrounds" does not have a single agreed-upon definition. Linguistically, "surrounds" is understood to mean a surrounding object (or objects) forms a barrier separating a surrounded object from the external space [Jackendoff 1976; Talmy 1983]. However, "surrounds" remains a *vague* spatial relation, i.e., one where there exist borderline cases, for which it is not possible to say categorically whether a region is or is not surrounded. Hence, any precise definition of "surrounds" will necessarily fail to capture the full range of linguistic interpretations.

Despite this fundamental difference between "surrounds" and "contains," their similarities (i.e., that both involve one object being separated from the external space by one or more others) has led to these two spatial relations being defined together. Liu et al. [Liu et al. 2008] argue that the "contains" relation from the 4IM [Egenhofer and Franzosa 1991], which requires multi-valued space (i.e., space where regions can overlap), has "surrounds" as its equivalent in a single-valued space (i.e., partitioned space, where regions cannot overlap). The idea that "contains" requires the interiors of regions to intersect, while "surrounds" requires that they do not, is a common distinction between these relations.

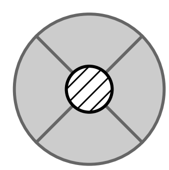
Another factor considered by most definitions of "surrounds" is whether the surrounded object is *partially* or *fully* enclosed. For example, an island "surrounded" by ocean is completely enclosed; a town "surrounded" by lakes is not. This distinction has similarities with the relations "geometrically inside" (partial enclosure) and "topologically inside" (full enclosure) proposed by Randell et al. [Randell et al. 1992] and expanded on by Cohn et al. [Cohn et al. 1997] (see Figure 1a). Geometric insideness (partial enclosure) allows the possibility of placing a region that overlaps the complement of the surrounding region's convex hull, without overlapping the surrounding region whereas topological insideness (full enclosure) means that there is no way for the smaller region to "escape" the larger region without passing through it.



(a) Geometric (left) and topological (right) insideness using the convex hull of the outer region (dashed lines).



(b) The convex hull of a set of regions (dashed lines) may run counter to human intuition about "surrounds". The center region is surrounded by the two outer regions.



(c) *surroundsAttach* relation between black striped inner and shaded grey outer regions after [Dube and Egenhofer 2014].

Fig. 1. Examples of surrounds and related concepts in the literature.

In this vein, a host of existing work has used the convex hull as the basis for constructing "surrounds" relations [Randell et al. 1992; Cohn et al. 1997; Donnelly 2005; Schultz et al. 2006; Bittner et al. 2008; Hahmann and Brodaric 2013; Bennett et al. 2013]. In these works, a larger region surrounds a smaller region if the smaller region is topologically inside the larger region's convex hull, and the two regions do not overlap.

The convex hull has also been applied to sets of regions that together "surround" another region (i.e., where the convex hull of the *set* of surrounding regions contains the surrounded region, though the surrounding and surrounded regions do not overlap) [Hahmann and Brodaric 2013]. Such a definition, however, does lead to cases that seem to diverge from our human intuition of "surrounds," such as that shown in Figure 1b. Further, while this definition can answer the decision problem "Is region x surrounded by the set of regions Y?" it does not necessarily lead to an efficient solution to the corresponding function problem: "Which subset Y (of all regions Z) surround region x?"

A further example of surrounds relations is Dube and Egenhofer's [Dube and Egenhofer 2014] *surroundsAttach* specifically for *partitioned* spaces (i.e., where regions are pairwise disjoint and jointly exhaustive of the space, see Figure 1c). Their *surroundsAttach* definition utilizes the adjacency graph of the partition's regions to determine for a given region the set of regions that, if removed from the adjacency graph, would disconnect that region from the partition's exterior. Because this definition is specific to a partitioned space, it detects only instances of full enclosure, as there can be no instances of partial enclosure. In cases of "non-partitioned" pairwise disjoint but not jointly exhaustive regions, Dube and Egenhofer [Dube and Egenhofer 2014] also provide the *surroundsDisjoint* definition, where a set of surrounding regions together fully (i.e., topologically) enclose a surrounded region.

Our approach is closest in spirit to that of Dube and Egenhofer [Dube and Egenhofer 2014]. We are concerned primarily with partial (geometric) enclosure for binary regions, i.e., pairwise disjoint sets of regions that are not jointly exhaustive of the space, so individual regions may not touch or overlap (such as, for example, the regions of oil/non-oil formed by an oil slick). We define three mutually exclusive types of relations a region can be:

— engulfed, when *partially* enclosed by one other region;
— surrounded, when *partially* enclosed by multiple regions; as well as for completeness
— enveloped, when *fully* enclosed by a single region with a hole.

While these definitions provide a new and consistent way of defining a range of qualitative region enclose relations, as we shall see they do naturally have correspondences to the existing definitions in the literature discussed above.

3. SIMPLE REGIONS

In this section, all regions are located within a finite rectangular frame, F, in the Euclidean plane. Let R be a collection of pairwise disjoint simple regions, each homeomorphic to a disk, contained in F. For each region $\mathbb{r} \in R$, the Voronoi region, $V(\mathbb{r})$, is the set of points in F closer to $\mathbb{r}$ than to any other region in R. $V(R)$ denotes the set $\{V(\mathbb{r}) : \mathbb{r} \in R\}$. Figure 2 shows a collection of regions $\mathbb{a}$–$\mathbb{h}$ (shown in grey) along with the boundaries of their Voronoi regions.

The boundaries of $V(R)$ constitute a planar embedded graph, $G(R)$. We use a *planar maptree* to formally represent $G(R)$. Maptrees have been introduced and documented in previous work by Worboys [Worboys 2012; Worboys 2013]. Here we give a brief description, using as example, $G(R)$ in Figure 3, where we have given arbitrary direction and labels for its edges. We note that $G(R)$ has three connected components: X, Y, and Z.

A *planar maptree*, M, is a rooted tree such that:

(1) nodes of M are alternately colored white and black, with the root colored white; and
(2) edges of M are labeled by strings.

A maptree, M, then represents a planar embedded graph G as follows:

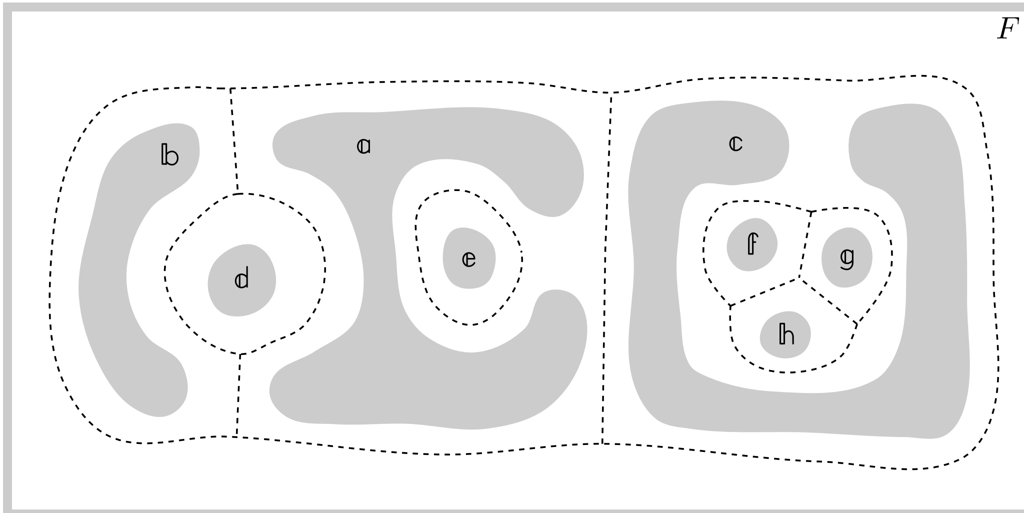


Fig. 2. Regions R (labeled 𝕒–𝕙, grey fill) and boundaries of corresponding Voronoi regions $V(R)$ (black lines)

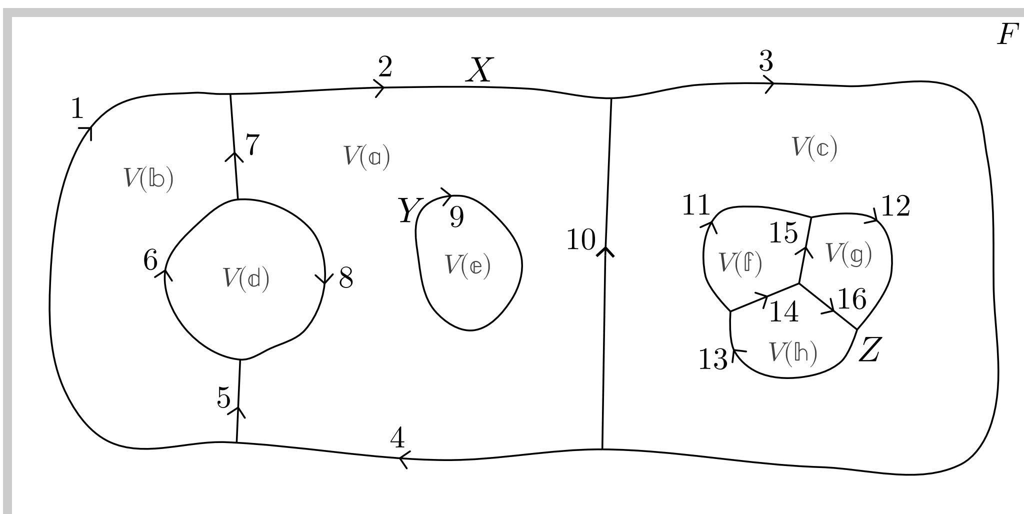


Fig. 3. Planar embedded graph $G(R)$

(1) Each white node represents a face of G, and the root node represents the outer face.
(2) Each black node represents a connected component of G. Each edge incident with specific black and white nodes is labeled with a string of symbols indicating the cycle of edges of the boundary represented by the black node around the face represented by the white node. Here, we use the convention that the direction of travel around a face boundary is anticlockwise, keeping the face on the left, with travel going in the opposite direction to a directed edge i indicated by $\bar{i}$.

A fuller description is given in [Worboys 2012; Worboys 2013], where it is shown that this representation is unique, up to continuous deformation. The maptree $M(R)$ for graph $G(R)$ shown in Figure 3 is shown in Figure 4. The three black nodes correspond to $G(R)$'s connected components X, Y, and Z.

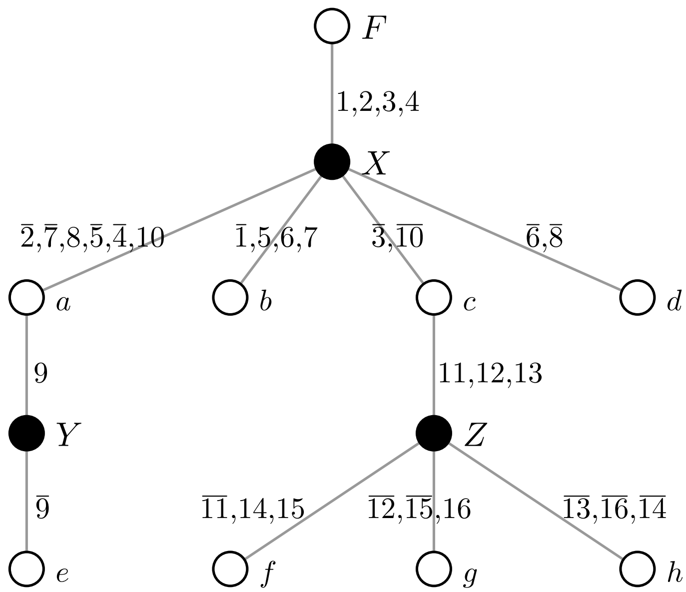


Fig. 4. Maptree $M(R)$

The maptree provides the solid formal basis for the spatial relations, "envelops," "engulf," and "surrounds," defined and computed in this paper. In the sequel, we will use consistently the notation:

— $\mathbb{r} \in R$ for a region and $V(\mathbb{r})$ for its Voronoi region (e.g., $\mathbb{d} \in R$ and $V(\mathbb{d})$ in Figures 2 and 3);
— $r \in M(R)$ for a (white) node in the maptree, corresponding to Voronoi region $V(\mathbb{r})$ (e.g., $d \in M(R)$ and $V(\mathbb{d})$ in Figures 3 and 4);
— $Bw \in M(R)$ for pairs of adjacent black and white nodes to refer to (undirected) edges in the maptree (e.g., $Xd \in M(R)$ in Figure 4);
— *label*(Bw) to retrieve the cycle that labels edge Bw in the maptree (e.g., *label*$(Xd) = (\overline{6}, \overline{8})$ from Figure 4);
— $z \in$ *label*(Br) for a directed edge z in the cycle that labels maptree edge Br (e.g., $\overline{6} \in$ *label*(Xd) from Figure 4); and
— *level*(n) to denote the length of the path from the root to a (black or white) node in the maptree $n \in M(R)$ (e.g., *level*$(Y) = 3$ in Figure 4).

Looking at the region configuration of Figure 2, one might intuitively say that region $\mathbb{d}$ is "trapped" by regions $\mathbb{a}$ and $\mathbb{b}$. Similarly, one might say that region $\mathbb{e}$ is "encircled' by region $\mathbb{a}$, or that regions $\mathbb{f}$, $\mathbb{g}$, and $\mathbb{h}$ are "hemmed in" by region 3. Using the maptree, we can more precisely capture these types of intuitive relations between regions. More specifically, we define two relations: *engulfs*, where a region is partially enclosed by a single region and *surrounds* when a regions is partially enclosed by multiple regions. As we shall see, a region may not be both engulfed and surrounded.

3.1. Engulfs

Informally, a region $\mathbb{r}$ is *engulfed* by region $\mathbb{s}$ if node s is an ancestor precisely two levels higher than node r, and the boundaries of $V(\mathbb{r})$ and $V(\mathbb{s})$ are directly adjacent.

Boundary adjacency can be deduced from maptree edge labels. In order to compare the labels of different maptree edges, we define $\odot$ as the *match* function, where $\textit{label}(Br) \odot \textit{label}(Br')$ represents the set of matching directed edges from the graph $G(R)$ between two labels (cycles) $\textit{label}(Br)$ and $\textit{label}(Br')$, i.e., $\{z_i, \bar{z}_i | z_i \in \textit{label}(Br), \bar{z}_i \in \textit{label}(Br') \textit{ or } z_i \in \textit{label}(Br'), \bar{z}_i \in \textit{label}(Br)\}$. In brief, the match function enables us to identify from the maptree the directed (embedded) edges in $G(R)$ shared by two Voronoi boundaries. For example, in Figure 4 $\textit{label}(Xb) \odot \textit{label}(Xd) = (\bar{1}, 5, 6, 7) \odot (\bar{6}, \bar{8}) = \{6, \bar{6}\}$, meaning that $V(\mathbb{b})$ and $V(\mathbb{d})$ share the directed edge pair 6, $\bar{6}$.

Combining the match function with the level function it is possible to define the engulfs relation as follows:

Definition 3.1. Region $\mathbb{r}$ is *engulfed by* $\mathbb{s}$ if, for the corresponding maptree nodes r and s, $\textit{level}(s) < \textit{level}(r)$ and there exists a (black) node $B \in M(R)$ such that $Br, Bs \in M(R)$ and $\textit{label}(Br) \odot \textit{label}(Bs) \neq \varnothing$.

In Figure 2 and its corresponding maptree in Figure 4, region $\mathbb{e}$ is engulfed by region $\mathbb{a}$ because their Voronoi regions share the pair of directed edges 9 and $\bar{9}$ in $G(R)$ through connected component Y. Similarly, regions $\mathbb{f}$, $\mathbb{g}$, and $\mathbb{h}$ are each engulfed by region $\mathbb{c}$. Region $\mathbb{d}$, by contrast, is not engulfed by any regions.

3.2. Surrounds

Engulfs describes the situation where a region is partially enclosed, but not fully contained, by a single region. In other cases, we can often find multiple regions that *together* partially enclose another region. We call this relation "surrounds," defined as follows:

Definition 3.2. Consider a white node $w \in M(R)$. If w has an ancestor $v \in M(R)$, such that $Bw, Bv \in M(R)$, $\textit{level}(v) < \textit{level}(w)$, and $\textit{label}(Bw) \odot \textit{label}(Bv) = \varnothing$ then the corresponding region $\mathbb{w}$ is said to be *surrounded*.

From Figure 2 and its corresponding maptree in Figure 4, region $\mathbb{d}$ is surrounded because there are no regions at a higher level of maptree that share a Voronoi boundary. While this definition is sufficient to indicate that a region is surrounded, to determine the set of surrounded regions, we must form an additional definition:

Definition 3.3. We define a binary relation $\sim$ on nodes in $M(R)$ such that $\forall x, y \in$M(R)$, x \sim y$ iff i. $\textit{level}(x) = \textit{level}(y)$; ii. there exists a (black) node $B \in M(R)$ where $Bx, By \in M(R)$; iii. $\textit{label}(Bx) \odot \textit{label}(By) \neq \varnothing$; iv. $x \neq y$.

Definition 3.4. Let $\mathbb{r}$ be a region that is surrounded in the sense of definition 3.2. Then $\mathbb{r}$ is said to be *surrounded by* the unique set of regions $\{\mathbb{t}_1, ..., \mathbb{t}_n\}$ corresponding to the set of nodes $\{t_i \in M(R) | t_i \sim r\}$.

Using this definition, region $\mathbb{d}$ is surrounded by regions $\mathbb{a}$ and $\mathbb{b}$ as they share the directed edges $8, \bar{8}$ and $6, \bar{6}$ respectively through connected component X.

3.3. Properties of engulfs and surrounds

We note that a region may not be both surrounded and engulfed. Specifically, for a surrounded region $\mathbb{r}$ and its corresponding maptree node r, definition 3.2 requires that for the (unique) parent black node B and grandparent white node w of r, edges Br and Bw in the map tree share no matching labels. By contrast, for an engulfed region $\mathbb{r}$, definition 3.1 requires that edges Br and Bw in the map tree positively do share some matching labels. Of course, a region may be neither engulfed nor surrounded (such as regions 𝕒, 𝕓, and 𝕔 in Figure 2 contained within the frame F, itself not a region).

From the literature already reviewed, we also note that our "engulfs" relation corresponds directly to the "surrounds" relations in [Randell et al. 1992; Cohn et al. 1997; Donnelly 2005; Schultz et al. 2006; Bittner et al. 2008; Hahmann and Brodaric 2013; **?**]. The "surrounds" relation in definition 3.3 has an indirect correspondence to the *surroundsAttach* relation of Dube and Egenhofer [Dube and Egenhofer 2014]. In short, a region 𝕣 that is surrounded by a set of regions $\{\mathbb{s}_1, \mathbb{s}_2, ...\}$ if the set of Voronoi regions $\{V(\mathbb{s}_1), V(\mathbb{s}_2), ...\}$ *surroundsAttach* the Voronoi region $V(\mathbb{r})$.

In distinguishing between the one-to-one relationship "engulfs" (also called "surrounds" in [Randell et al. 1992; Cohn et al. 1997; Donnelly 2005; Schultz et al. 2006; Bittner et al. 2008; Hahmann and Brodaric 2013; **?**]) and the many to one relationship "surrounds" (akin to *surroundsAttach* relation of Dube and Egenhofer [Dube and Egenhofer 2014]) we have preferred the natural language interpretation of of a police force "surrounding" a building, reserving "engulfs" echoing the geographical interpretation of a "gulf" (as in the Gulf of California, the sea area engulfed by the west coast of Mexico).

4. GENERAL REGIONS

While the methods of the previous section are sufficient for simple regions, minor modifications are required to extend these definitions to accommodate general regions that may have holes and in which may be nested other regions. While we assume that such general regions are again pairwise disjoint and located within a finite rectangular frame, they may not now be homeomorphic to a disk. In capturing general region configurations, it is necessary to generate Voronoi regions not only of the "positive" regions, $V(R^+)$ (the regions themselves, as for simple regions), but also of the "negative" regions, $V(R^-)$ (i.e., of the holes and spaces between regions). An example configuration of general regions is shown in Figure 5, along with the boundaries of their Voronoi regions.

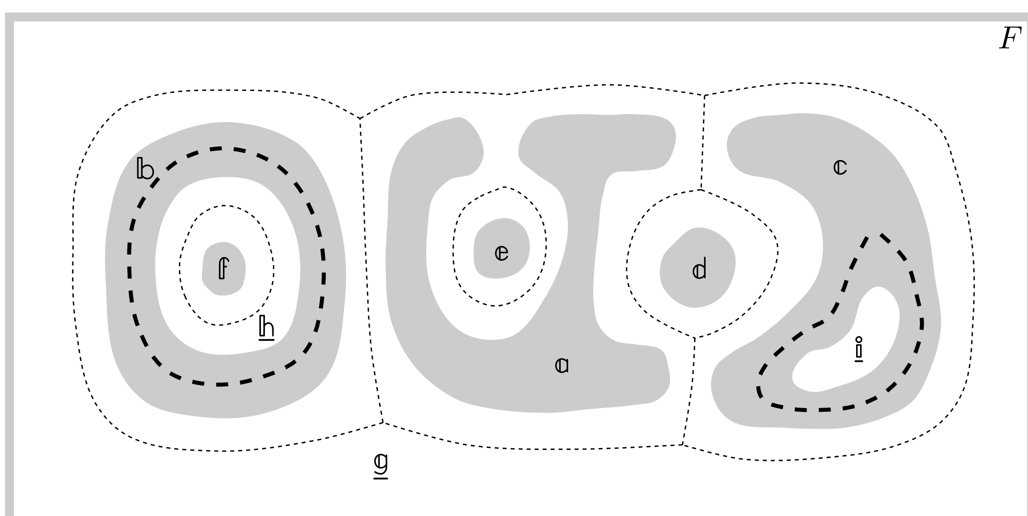


Fig. 5. General regions example. Positive regions R^+ (labeled 𝕒–𝕗, grey fill), negative regions R^- (underlined and labeled 𝕘–𝕚, white fill), and boundaries of corresponding Voronoi regions $V(R)$ (thin and thick lines respectively)

The boundaries of $V(R^+)$ and $V(R^-)$ now constitute two planar embedded graphs, $G(R^+)$ and $G(R^-)$, as shown in Figure 6, derived from Figure 5.

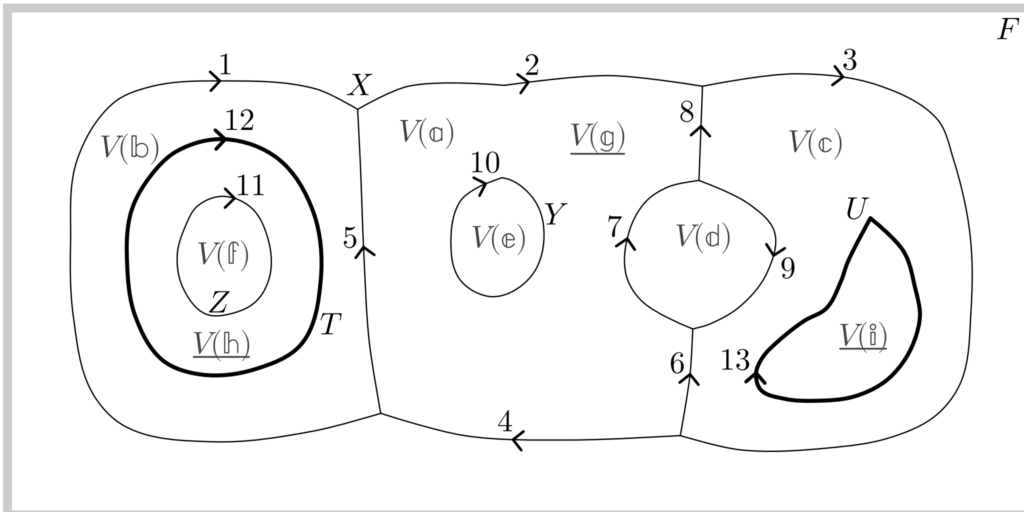


Fig. 6. Planar embedded graphs of positive and negative regions, $G(R^+)$ and $G(R^-)$. For clarity, negative Voronoi region labels have been underlined.

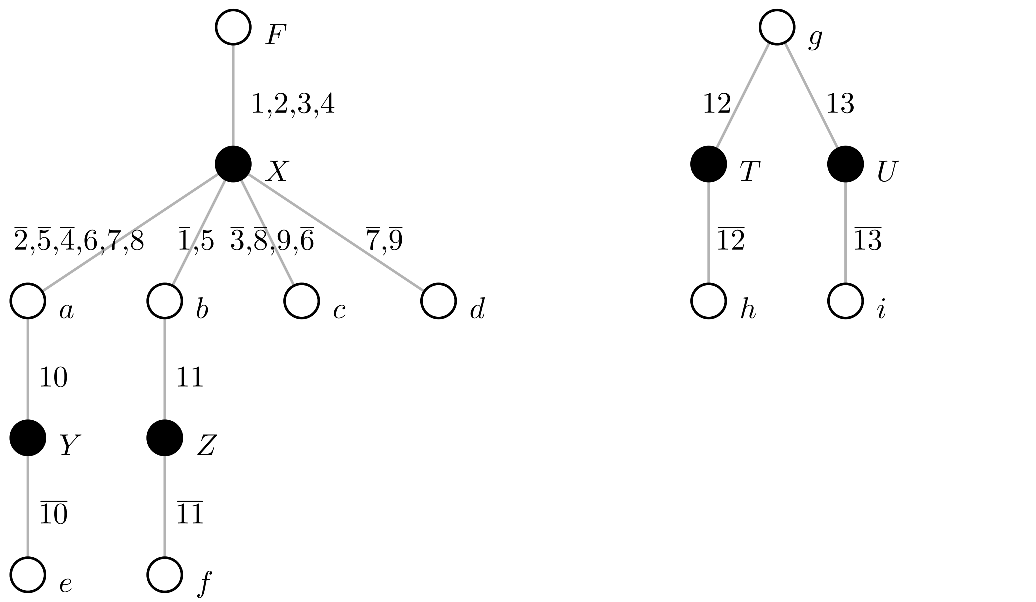
4.1. Engulfed, surrounded, and enveloped

As in the simple region example, the structure of these two sets of Voronoi regions, $V(R^+)$ and $V(R^-)$, can each be represented using maptrees, in this case $M(R^+)$ and $M(R^-)$ for the positive and negative regions respectively. The mapping between regions, Voronoi regions, and maptree nodes remains unchanged. For example, each positive region $\mathbb{r} \in R^+$ corresponds to a (white) node $r \in M(R^+)$ which represents the Voronoi region $V(\mathbb{r})$. Edges in the maptrees are again referenced using their end nodes, e.g., $Br \in M(R^+)$.

The general regions in Figure 5 again provide examples of our "surrounds" relation. For example, in the region configuration (Figure 5) and the positive maptree (Figure 7a), the (positive) region $\mathbb{d}$ is surrounded by the (positive) regions $\mathbb{a}$ and $\mathbb{c}$ (see definition 3.3). However, looking only at the positive maptree and definition 3.1, it might appear that, for example, region $\mathbb{e}$ is engulfed by the region $\mathbb{a}$; and region $\mathbb{f}$ is engulfed by region $\mathbb{b}$. Turning instead to region configuration in Figure 5, it is clear that region $\mathbb{e}$ is indeed partially enclosed by the region $\mathbb{a}$ (i.e., true engulfs), but that region $\mathbb{f}$ is fully enclosed by region $\mathbb{b}$.

To accommodate this, general regions require an amended definition 3.1 to distinguish between true "engulfs" and full enclosure, which relation we term "envelops." Envelopment in the positive map tree is indicated by the presence of a connected component in the negative maptree, where the corresponding (negative) Voronoi boundary that spatially contains enveloped region's (positive) Voronoi boundary.

Envelopment therefore requires the combination of information from both positive and negative maptrees, related using the *map* function, such that:



(a) Maptree representing the positive regions, $M(R^+)$.

(b) Maptree representing the negative regions, $M(R^-)$.

Fig. 7. Maptree example based on general regions from Figure 5.

$$\mathit{map}(B) = \begin{cases} r \in M(R^+) \text{ s.t. region represented by } B \text{ is inside } V(\mathbb{r}) \text{ if } B \in M(R^-) \\ r \in M(R^-) \text{ s.t. region represented by } B \text{ is inside } V(\mathbb{r}) \text{ if } B \in M(R^+) \end{cases}$$

We may note that r is always unique in the above definition.

Using the example in Figures 6 and 7, *map*$(Z) = h$ captures that the Voronoi boundary represented by connected component Z is spatially contained within the Voronoi region $V(\mathbb{h})$ generated by region $\mathbb{h}$. Likewise, it can be seen that connected components X and Y are within Voronoi region $V(\mathbb{g})$ (*map*$(X) = g$, *map*$(Y) = g$), and T and U are within Voronoi regions $V(\mathbb{b})$ and $V(\mathbb{c})$ respectively (*map*$(T) = b$, *map*$(U) = c$). Using this *map* function, it is now possible to distinguish the engulfs and envelops relations as follows:

Definition 4.1. Consider nodes r, s, $B \in M(R^+)$ such that Br, $Bs \in M(R^+)$, *level*$(s) <$ *level*(r), and *label*$(Br) \odot$ *label*$(Bs) \neq \varnothing$, then region $\mathbb{r}$ is either *engulfed* or *enveloped by* $\mathbb{s}$. If there exists $B_2 \in M(R^-)$ such that $(B_2, \mathit{map}(B)) \in M(R^-)$, *level*$(B_2) <$ *level*$(\mathit{map}(B))$, and *map*$(B_2) = s$, then $\mathbb{r}$ is *enveloped by* $\mathbb{s}$. Otherwise $\mathbb{r}$ is *engulfed by* $\mathbb{s}$

For example, region $\mathbb{b}$ envelops (but does not engulf) region $\mathbb{f}$ due to the presence of connected component T within Voronoi region $V(\mathbb{b})$. Region $\mathbb{e}$, in contrast, is engulfed (and not enveloped) by region $\mathbb{a}$.

5. DECENTRALIZED ALGORITHM

Identifying engulfs, surrounds, and envelops relationships within a centralized spatial information system, such as a GIS or spatial database, requires only the straightforward application of common geometric functions. In this section, however, we show how these relationships can also be efficiently identified by a decentralized algorithm. This algorithm is suited to highly distributed computing environments, such as geosensor networks. In order to ensure our algorithm is adaptable to the widest possible range of computing environments, we make the minimum possible assumptions about the capabilities of the geosensor network. Specifically, we assume that:

— nodes may move (with static nodes a special case of movement, with zero speed);
— nodes have no access to coordinate positioning information.

We further assume that for a set of mobile nodes V connected by a time-varying communication network $G(t) = (V, E(t))$:

— nodes are uniquely identifiable (formally, we assume an *identifier function*, $id : V \to \mathbb{N}$);
— nodes are aware of and are able to communicate with their current neighbors (captured as a *neighborhood function*, $nbr : V \times T \to 2^V$, where $nbr(v,t) \mapsto \{n \in V | \{n, v\} \in E(t)\}$); and
— nodes are equipped with sensors able to detect their immediate environment, abstracted as simply sensing a positive or negative region (represented as a *sensor function* $s : V \times T \to \{0, 1\}$).

Although our specification is abstracted, such nodes can be imagined sensing information about the underlying geographic regions in a geographic space, such as hot spots, presence of specific pollutants, or indeed any environmental variable. Movement itself might be self-powered (such as robotic nodes) or with nodes either static or attached to mobile agents, such as vehicles, people, or animals. It should be noted that while the nodes may be mobile, the regions they sense are assumed to be static.

5.1. Algorithm overview

The algorithm design can be broken up into five modules based on function, with the relations between these modules shown in Figure 8. In brief, the five modules work as follows:

(1) **Region identification:** Using leader election [Santoro 2006], the network is initialized to create and distribute unique identifiers for each positive and negative region (rid) in the network.
(2) **Voronoi region identification:** Using hop-count flooding, the approximate boundaries of the Voronoi regions induced by the positive and the negative regions is determined. By storing the hop count to each adjacent region in their boundary table (B_t), nodes recognize they are at the boundary of two Voronoi regions if the hop counts from two neighboring regions are equal. Nodes also set their adjacent region id (adj) to the region id of the record with the shortest hop count in their boundary table. As the nodes are mobile, this must be refreshed periodically. This approach is similar in concept to [Malazi et al. 2013], which uses mass-based diffusion to “push” out and maintain approximate Voronoi boundaries between a set of points in a mobile geosensor network.
(3) **Voronoi boundary propagation:** Using the boundary table (B_t), nodes on a boundary between Voronoi regions will add that boundary to their maptree table (M_t). Surprise flooding is then used to propagate these boundaries throughout the network.
(4) **Maptree generation:** Once the maptree timer elapses, each entry in the maptree table is assigned a connected component. This timer is set to elapse at a set

period after module 3 has completed so that every node has the complete set of information necessary to construct the maptree.

(5) **Node movement:** When a node enters a new region it must refresh its boundary table (B_t) and swap its region and Voronoi region ids.

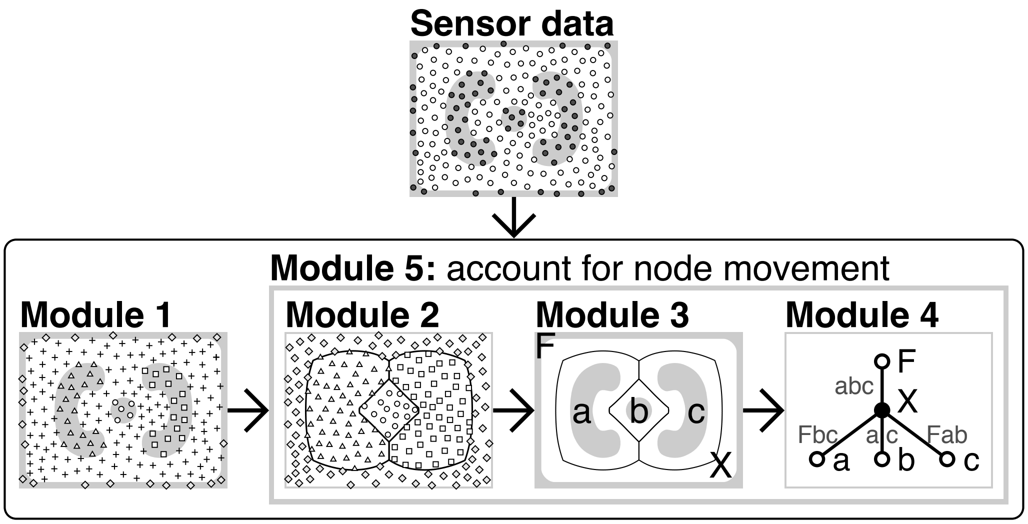


Fig. 8. Flow diagram representing the interactions between the modules that comprise the decentralized algorithm.

Our algorithms build on previous work in [Both and Duckham 2013], which proposed a decentralized algorithm for determining containment and adjacency relations between regions. While modules 1 and 2 remain unchanged, the other modules are extended here to determine whether a region is engulfed or surrounded (rather than strictly contained) by other regions.

The algorithms are implemented based on the specification-style of Santoro [Santoro 2006] and extended in [Duckham 2012] where node behavior and interactions are defined using the components: restrictions (environmental assumptions); events (when messages are received or sensors triggered); actions (responses to events); and states (where nodes may perform different actions in response to the same event). To reduce ambiguity, we make the distinction between global functions, which use no notation (e.g., id); local versions of these functions, which use overdot notation (e.g., $\mathring{id}$); and received outputs of these functions and variables, which use prime notation (e.g., id').

Due to the sparse nature of the data collected by the decentralized algorithms, unique edge labeling in the planar embedded graph is not preserved. Instead the algorithms only record whether a boundary is present between two Voronoi regions. As unique edge identification is not possible, the cycles that label maptree edges (i.e., $\mathit{label}(Br)$) will be replaced with the set of nodes $r_i \in M(R)$ corresponding to Voronoi regions that share a boundary with the Voronoi region $V(r)$ through connected component B, i.e., $\mathit{label}(Br) \mapsto \{r_i \in M(R) | \mathit{label}(Br) \odot \mathit{label}(Br_i) \neq \varnothing\}$. For example in Figure 4, the cycle of maptree edge Xa, $(\overline{2}, \overline{7}, 8, \overline{5}, \overline{4}, 10)$ will be replaced with the set $\{F, b, c, d\}$. While this does lead to a maptree with reduced information content, unique edge identification is not required for the relations to be deduced.

As all modules bar module 4 are constructed from small modifications to well known decentralized algorithms, it is only module 4 that will be discussed in detail. Using Figure 2 as an example, module 4 splits the Voronoi regions into connected components (shown as black lines in Figure 3). First, the boundary pairs from each entry in the Maptree table (M_t) are added to the *boundaries* collection (Line 4), which will be assigned to connected components until no records remain (Lines 7–16). If there are no entries in the *group* set, one of the *boundaries* entries will be moved to the *group* set, e.g., $\{F, a\}$ (Lines 8–10).

Module 4 Maptree generation

1: Restrictions: reliable communication; connected, bidirected communication graph $G(t) = (V, E(t))$; identifier function $id : V \to \mathbb{N}$; neighborhood function $nbr : V \times T \to 2^V$; sensor function $s : V \times T \to \{0, 1\}$; connected component labeling function: $clabel(c) \to \mathbb{N}$ where $\mathbb{N}$ is a unique id for that set of regions.

2: Local variables: region id $rid : V \to \mathbb{N} \cup \{-1\}$, initialized to $\mathring{rid} := \mathring{id}$; sensor function $s : V \to \{0, 1\}$; adjacent region id $adj : V \to \mathbb{N} \cup \{-1\}$, initialized to $\mathring{adj} := -1$; boundary table $B_t = \langle \boldsymbol{bid} : \mathbb{N}, \boldsymbol{h} : \mathbb{N}\rangle$, initialized with zero records; maptree table, $M_t = \langle rid_a : \mathbb{N}, rid_b : \mathbb{N}, cid : \mathbb{N}\rangle$, initialized with record $(-1, root, \varnothing)$ where $root$ is the id of the region representing the exterior.

REGN

3: *When* maptree timer elapsed
4:   **let** *boundaries* := *collection of sets derived from entries from* M_t
5:   **let** *components* := *collection of sets, initialized to* $\varnothing$
6:   **let** *group* := $\varnothing$
7:   **while** *boundaries* $\neq \varnothing$ **do**
8:     **if** *group* $= \varnothing$ **then**
9:       **set** *group* := ONE OF *boundaries*
10:       **set** *boundaries* := *boundaries* $\setminus$ *group*
11:     **if** i exists s.t. $\{i, j\} \in$ *boundaries* AND $\{i, k\} \in$ *boundaries* AND $j, k \in$ *group* **then**
12:       **set** *group* := $i \cap$ *group*
13:       **set** *boundaries* := *boundaries* $\setminus \{\{i, j\}, \{i, k\}\}$
14:     **else**
15:       **if** *group* $\neq \varnothing$ **then**
16:         **set** *components* := *components* $\cap$ *group*
17:   **for all** $l \in M_t$ **do**
18:     **for all** $m \in$ *components* **do**
19:       **if** $\{l.rid_a, l.rid_b\} \subseteq m$ **then**
20:         UPDATE l SET $cid = clabel(m)$

The *group* set then attempts to find two boundary records that share one region between themselves and one region each with the *group* set (Line 11). For example, records $\{a, c\}$ and $\{F, c\}$. The common region is then added to the *group* set: (i.e., $\{F, a, c\}$) and the suitable records are removed from the *boundaries* collection (Lines 12–13). This continues on until no more suitable records can be found and the completed group $\{F, a, b, c, d\}$ is added to the *components* collection (Lines 14–16). The *components* collection $\{\{F, a, b, c, d\}, \{a, e\}, \{c, f, g, h\}\}$ is the result of processing all *boundaries* entries.

Lastly, every entry of the maptree table is sorted through to find each region's associated connected component (Lines 17–18). This will be the connected component that contains both region ids from the maptree table's entry (Line 19). This record's cid field will then be updated with the label of the connected component (Line 20). This label is obtained using the *clabel* function, which determines a unique id for a set of region ids. For maptree table entries $(F, a, \varnothing)$ and $(g, h, \varnothing)$ this would involve finding which connected component contains both region ids. In this case, that would be X and Z, creating the entries (F, a, X) and (g, h, z).

Given that the detection of the surrounds and engulfs relations rely on a level function, the id of the root region (i.e, the exterior region) must be known. This can be done by either having a node on the outermost region use surprise flooding to distribute the root node id throughout the network, or for the network to simply be initialized knowing the root. For consistency, we set the root node to F. Once the maptree table has been completed, it can, along with the root node, be used to determine any surrounds or engulfs relations.

5.1.1. General algorithm. To extend the simple algorithm so that it is capable of working with general regions, some key changes must be made. Specifically, a map table ($P_t = \langle cid : \mathbb{N}, rid : \mathbb{N} \rangle$) must be added to record the regions the connected components reside within, which will enable use of the *map* function to identify envelops relations. Additionally, a *neg* column is added to the maptree table (M_t) to indicate whether the entry is part of the maptree storing positive regions (*neg* $= 0$) or the maptree storing negative regions (*neg* $= 1$).

Module 3 runs in essentially the same way as its simple counterpart, with the exception that the maptree table is restricted to nodes within the regions that detected them. When complete, module 4 then fills the maptree table's neg column with the nodes' sensed value. Directly adding the sensed value works as nodes in negative regions detect segments of positive maptrees ($M(R^+)$), and nodes in positive regions detect segments of negative maptrees ($M(R^-)$). As all the connected components presently stored within a node have originated from that node's current region, the map table (P_t) can be populated with these records and the region id, rid. By segmenting the maptree in this way, it is possible to reduce communication costs.

Given that each node is restricted to entries in the maptree and map tables that have originated from that node's current region, surprise flooding is used to propagate these entries throughout the network. This is to ensure that upon the completion of module 4, every node will have access to the information necessary to determine the presence of any "surrounds," "engulfs," or "envelops" relations.

For module 5, when a node changes regions and module 4 has not yet run, the maptree table is cleared and replaced with new records requested from the node's new neighbors. Nodes receiving this request will respond if the received sensed value matches the node's own (i.e., the message is from a node in the same region), and if the node has a non-empty maptree table (i.e., the node has not also just transitioned and thus cleared its variables). The node will then respond by sending a response message along with its sensed value and maptree table.

Nodes receiving this response will again check to see that the received sensed value matches the node's own. This is necessary for cases where there is both an arrival and departure of nodes from a region in close proximity (e.g., nodes moving from region $\mathbb{c}$ to $\mathbb{g}$ as well as from $\mathbb{g}$ to $\mathbb{c}$) as nodes could be updated with records from the wrong region. If the message is from the same region and the maptree table is empty, then it will be filled using records from the received message.

6. RESULTS

The ways in which decentralized algorithms are evaluated can be divided into two categories: scalability and veracity. Scalability is determined by how the amount of node communication is affected by an increase in network size, and is referred to as communication complexity or load balance when evaluating in terms of the entire network or individual nodes respectively.

Veracity tests the accuracy of decentralized algorithms by comparing the results of the algorithms with the actual results as determined by the environment. As stated previously, modules 1 and 2 originate from previous work [Both and Duckham 2013], in which their veracity was tested. As the algorithms of this work require correct information from these two modules to construct their specific formal models, meeting the necessary requirements for modules 1 and 2 to perform correctly will produce correct results for the entirety of the algorithms. In brief, previous work found that meeting the criteria of sufficient node density, broadcast interval, and communication distance produced accurate results, and that these three factors were dependent upon the characteristics of the phenomena being monitored. These findings were used when implementing the scalability experiments to ensure correct results were obtained.

As in previous work, the algorithms were implemented using the NetLogo agent-based simulation system [Wilensky 1999], with simulated objects moving according to a correlated random walk [Bartumeus et al. 2005] as it provides a simple approximation of many natural movement patterns [Ramos-Fernández et al. 2004; Sims et al. 2008].

The algorithms tested ran on either the simple region configuration of Figure 2 or the general region configuration of Figure 5. These configurations were discretized into a 290×145 grid, with an additional region added around the edge of the frame to serve as the root. Communication distance was initially set to a radius of ten grid cell widths, then reduced in proportion to network size to ensure the level of node connectivity remained consistent across network sizes.

In order to reduce the amount of communication necessary to run the algorithms, message aggregation was implemented. Nodes temporarily store all messages received during a tick (simulation timestep), remove any duplicate messages, and compare the remaining messages before processing them.

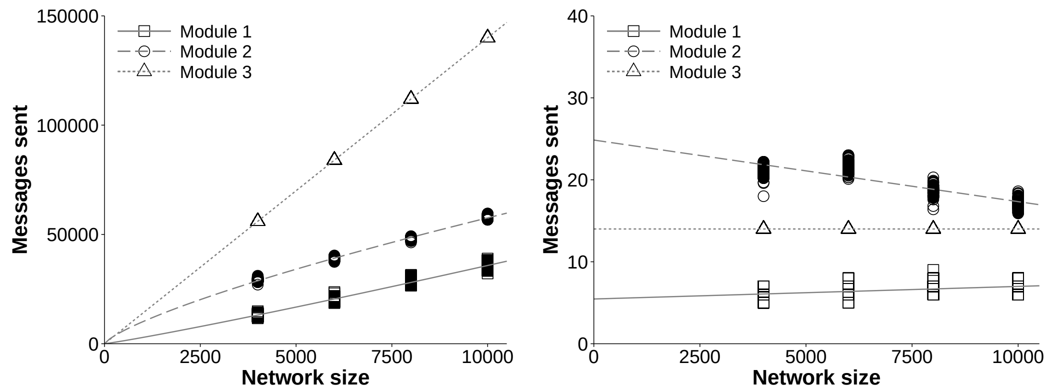
Node movement occurred every 50 ticks, with a movement distance of 2.5 grid cell widths and module 2 was rerun every 25 ticks. Module 3 was set to run after 170 ticks, and module 4 was set for 50 ticks later (at 220), giving module 3 time to complete before module 4 began. To improve graph readability, the number of messages sent by module 2 was divided by the number of broadcast rounds (i.e., by 10), meaning that the number of messages plotted for module 2 represents the average number of messages sent for a single broadcast round.

After 350 ticks, the total number of messages sent by each module for the entire network was recorded (i.e., communication complexity) as well as the maximum number of messages sent by an individual node for each module (i.e., load balance). This was done 100 times with randomized node positions for each network size (4,000, 6,000, 8,000, and 10,000 nodes), leading to a total of 400 experimental runs for each tested algorithm and region configuration. Regression curve formulas and R^2 values for the communication complexity can be found in Table I for every experiment conducted, which have all achieved a good fit, as evidenced by R^2 values of greater than 0.98. As modules 4 and 5 do not broadcast messages in the simple algorithm they have not been included.

6.1. Simple region configuration

Looking at the simple region configuration of Figure 2, approximately half of the area is covered by the nine positive regions (recall that an extra region has been added to the boundary of the frame), with the remainder covered by the single negative region. This region configuration consists of 14 unique Voronoi region boundaries.

Simple algorithm. Figures 9a and 9b show the communication complexity and load balance of the algorithm devised for simple regions. Modules 1 and 2 run identically for both algorithms, with module 1 being based on leader election. Leader election algorithms are typically of polynomial order, however module 1 has exhibited weakly polynomial growth, which is consistent with the slightly positive linear fit of its load balance.



(a) Communication complexity. (b) Load balance.

Fig. 9. Scalability of communication for the simple algorithm using a simple region configuration.

Modules 2 and 3 are flooding based algorithms, which are typically of linear order. While module 3 scaled linearly, sending exactly 14 messages per node (corresponding to the 14 unique Voronoi region boundaries), module 2 scaled sub-polynomially, which is consistent with the flat and slightly negative linear fits of their load balance. Both modules 1 and 2 have lower than expected order due to message aggregation causing proportionally greater messages to be discarded as network size increases.

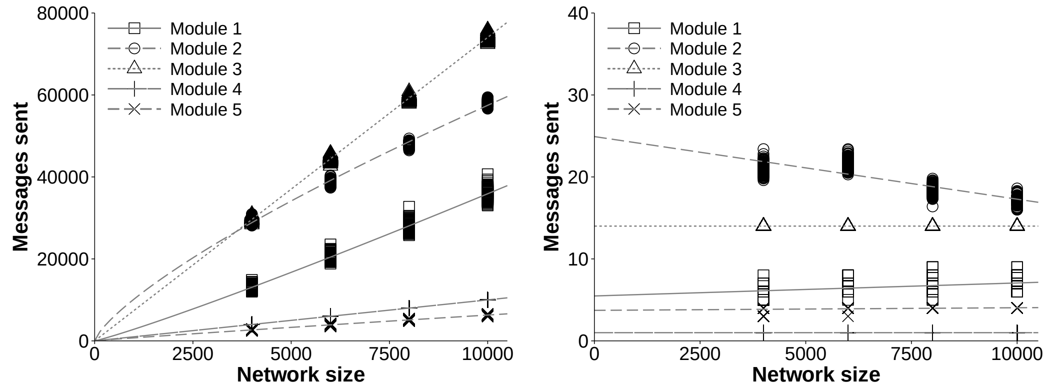
General algorithm. Module 4 is also based on flooding, and from Figure 10a, both modules 3 and 4 were inferred to scale linearly in terms of communication complexity, sending approximately 7.4 and exactly 1 message per node. This is consistent with the flat linear fits of their load balance (Figure 10b). For module 3, the 14 unique Voronoi region boundaries are only broadcast throughout the single negative region, which is occupied by approximately half the nodes. For module 4, each portion of the computed maptree must be broadcast, of which there is only one in this region configuration.

Module 5 is a request/response based algorithm, which typically scale linearly, although here was inferred to be of sub-polynomial order, consistent with the slightly negative linear fit of the load balance. Similar to module 2, message aggregation has again resulted in sub-polynomial growth.

6.2. General region configuration

Looking at the general region configuration (Figure 5), approximately half of the area is covered by one of the seven positive regions, with the remainder covered by one of the three negative regions. This region configuration consists of 11 unique Voronoi region boundaries, with nine of these boundaries located within the negative regions, and two located within the positive regions.

General algorithm. Figures 11a and 11b show the communication complexity and load balance of the general algorithm running on a general region configuration. While all modules were found to be of the same order communication complexity, modules 1–3

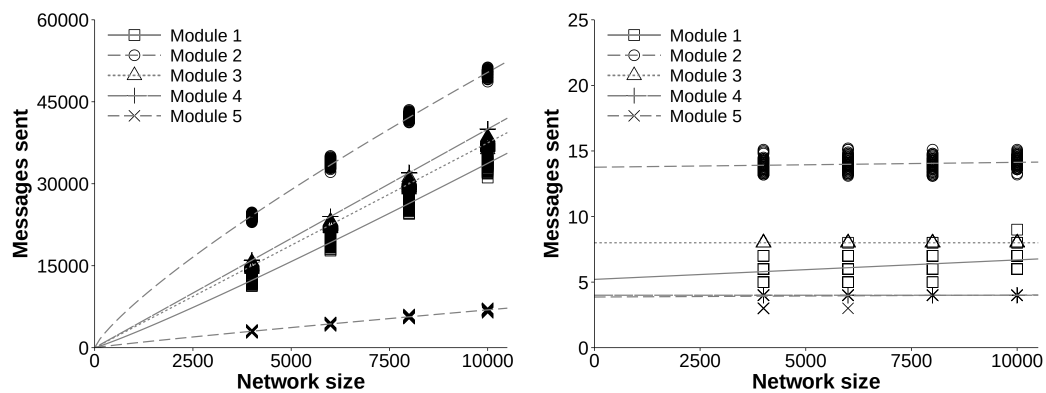


(a) Communication complexity.

(b) Load balance.

Fig. 10. Scalability of communication for the general algorithm using a simple region configuration.

sent overall fewer messages for the general than the simple region configuration. This reduction was due to the general region configuration having fewer Voronoi region boundaries and messages sent being restricted to overall smaller regions.



(a) Communication complexity.

(b) Load balance.

Fig. 11. Scalability of communication for the general algorithm using a general region configuration.

Module 4 increased to four messages sent per node as there are four portions of the computed maptree to be broadcast, as apposed to a single portion in the simple region configuration. Module 5 remained unchanged from the simple region configuration due to approximately the same proportion of nodes changing regions.

6.3. Summary

Table I shows a summary of the communication complexity of the algorithms tested on simple and general region configurations. From this table, it can be seen that all regression curves for each of the algorithm's modules achieved a good fit, as evidenced by R^2 values of greater than 0.98. Looking at the regression curves, only module 1 is of polynomial order. This is due to module 1's broadcasting characteristics being based on leader election algorithms. Modules 2–5 are instead based on surprise flooding and responses to nodes that are changing regions, leading to either linear or sub-polynomial order communication complexity.

Table I. Scalability of algorithms in terms of the total number of messages sent by each module. Regression curve for module 2 shows the number of messages sent per broadcast round.

			Simple regions		General regions	
		Module	Regression	R^2	Regression	R^2
Algorithms	simple	1.	$y = 1.513x^{1.09}$	0.9852		
		2.	$y = 54.213x^{0.76}$	0.9951		
		3.	$y = 14.000x$	1.0000		
	general	1.	$y = 1.347x^{1.10}$	0.9848	$y = 1.369x^{1.10}$	0.9899
		2.	$y = 56.346x^{0.75}$	0.9945	$y = 31.161x^{0.80}$	0.9970
		3.	$y = 7.405x$	0.9987	$y = 3.478x$	0.9983
		4.	$y = 1.000x$	1.0000	$y = 4.000x$	1.0000
		5.	$y = 1.140x^{0.94}$	0.9898	$y = 1.597x^{0.91}$	0.9907

It is important to note that while the number of messages sent differs between the simple and the general region configurations for the general algorithm, the order of the modules' scalability is unchanged. This consistency indicates that region configuration does not affect the scalability of the algorithms.

Given that modules 1 and 2 are the same for all algorithms, these two modules can be ignored when comparing overall scalability. For the simple region configuration, the simple and general algorithms sent an average of exactly 14 and approximately 9.1 messages per node. The superior performance of the general algorithm is due to the confinement of its module 3 messages to individual regions. It is clear from this that computing segments of the maptree in individual regions and then broadcasting the completed segments throughout the network has lead to gains in efficiency.

7. CONCLUSIONS AND FURTHER WORK

This work has investigated the monitoring of both simple and general region configurations using two decentralized algorithms. It was found that all modules of the algorithms exhibited either sub-polynomial, linear, or weakly polynomial scalability (with the worst case being $O(n^{1.1})$). The order of scalability produced was due to the type of decentralized algorithm the module was based on, with leader election based algorithms producing weakly polynomial scalability, and surprise flooding algorithms producing sub-polynomial or linear scalability. For the efficiency of the algorithms, the general algorithm was found to perform better than the simple algorithm.

In addition to algorithm scalability, it is important to consider the costs of running the modules long-term, in particular module 2 which must be rerun periodically. Given that the nodes in a dynamic (mobile) geosensor network have limited battery capacity, such modules can only be run a certain number of times before the battery is depleted. In the interests of extending network run-time, it is therefore important to consider how often particular modules should be run.

While the experiments were constructed with sufficient node density, broadcast interval, and communication distance to produce accurate results, the values chosen also had implications for region granularity. Communication distance also represents the minimum distance between the edges of two potential regions for them to be considered separate regions, meaning that different granularities of sensors will detect different spatial relationships. While communication distance was chosen to always be smaller than the minimum distance between any two regions in Figures 2 and 5, many phenomena, such as oil spills and algal blooms, can be measured at multiple levels of granularity.

Network granularity also has implications for the cost of the network as communicating at larger distances requires more energy, but also requires fewer nodes to cover the same area. Assuming the same battery capacity, working at a coarse granularity allows for a network with fewer nodes but a shorter network lifespan whereas a fine granularity requires a network with more nodes but a longer network lifespan.

Unlike containment, surrounds is a far more vague concept to capture. As demonstrated by their corresponding spatial relations, containment is supported by a single crisp definition, while surrounds is not. When constructing a suitable definition for surrounds, it is clear from our intuition of "surrounding" that there are cases where one or more regions clearly surround another, and cases where they clearly do not. Any definition must therefore take a position between these two endpoints in a way that minimizes edge cases that run counter to this intuition.

Our goal here was to construct a set of mutually exclusive relations that together encode all of the concepts of surrounds in use in a way that was consistent with existing definitions in the literature. Specifically, this work developed three qualitative spatial relations based on the maptree, which are:

— *envelops*, where a single region fully encloses another region,
— *engulfs*, where a single region partially encloses another region, and
— *surrounds*, where multiple regions partially enclose another region.

Instead of storing the relations between regions, the maptrees featured in this work store Voronoi regions that are induced by the regions, of which there is a one-to-one mapping.

The long term focus of this work is to apply the maptree to cases where the regions being observed are dynamic. By extending the maptree to account for region dynamism, the specific way region configurations enter and exit the three qualitative spatial relations can be described using a conceptual neighborhood graph, providing a detailed description of how the relations between regions can change over time.

REFERENCES

Besim Avci, Goce Trajcevski, and Peter Scheuermann. 2014. Managing Evolving Shapes in Sensor Networks. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management (SSDBM '14)*. ACM, New York, NY, USA, 22:1–22:12. DOI:http://dx.doi.org/10.1145/2618243.2618264

Frederic Bartumeus, M. G E. da Luz, Gandhimohan M. Viswanathan, and Jordi Catalan. 2005. Animal Search Strategies: A Quantitative Random-Walk Analysis. *Ecology* 86, 11 (Nov. 2005), 3078–3087. DOI:http://dx.doi.org/10.1890/04-1806

Brandon Bennett, Vinay Chaudhri, and Nikhil Dinesh. 2013. A Vocabulary of Topological and Containment Relations for a Practical Biological Ontology. In *Spatial Information Theory (Lecture Notes in Computer Science)*. Springer, Scarborough, UK, 418–437. DOI:http://dx.doi.org/10.1007/978-3-319-01790-7_23

Thomas Bittner, Maureen Donnelly, Louis J. Goldberg, and Fabian Neuhaus. 2008. Modeling Principles and Methodologies - Spatial Representation and Reasoning. In *Anatomy Ontologies for Bioinformatics*, Albert Burger, Duncan Davidson, and Richard Baldock (Eds.). Number 6 in Computational Biology. Springer, 307–326. DOI:http://dx.doi.org/10.1007/978-1-84628-885-2_15

Alan Both and Matt Duckham. 2013. Qualitative Spatial Structure in Complex Areal Objects Using Location-Free, Mobile Geosensor Networks. In *2013 IEEE 13th International Conference on Data Mining Workshops*. IEEE, Dallas, TX, 978–985. DOI:http://dx.doi.org/10.1109/ICDMW.2013.157

Anthony G. Cohn, Brandon Bennett, John Gooday, and Nicholas Mark Gotts. 1997. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. *GeoInformatica* 1, 3 (Oct. 1997), 275–316. DOI:http://dx.doi.org/10.1023/A:1009712514511

Maureen Donnelly. 2005. Containment Relations in Anatomical Ontologies. In *AMIA Annual Symposium Proceedings*, Vol. 2005. AMIA, Washington, DC, 206–210. http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1560437/

Matthew P. Dube and Max J. Egenhofer. 2014. Surrounds in Partitions. In *Proceedings of the 22Nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '14)*. ACM, New York, NY, USA, 233–242. DOI:http://dx.doi.org/10.1145/2666310.2666380

Matt Duckham. 2012. *Decentralized Spatial Computing: Foundations of Geosensor Networks*. Springer. DOI:http://dx.doi.org/10.1007/978-3-642-30853-6

Matt Duckham, Stefan Dulman, Jörg-Rüdiger Sack, and Monika Sester. 2014. Geosensor Networks: Bridging Algorithms and Applications (Dagstuhl Seminar 13492). *Dagstuhl Reports* 3, 12 (2014), 17–42. DOI:http://dx.doi.org/doi.org/10.4230/DagRep.3.12.17

Max J. Egenhofer and Robert D. Franzosa. 1991. Point-set topological spatial relations. *International Journal of Geographical Information Systems* 5, 2 (Jan. 1991), 161–174. DOI:http://dx.doi.org/10.1080/02693799108927841

Max J. Egenhofer and John Herring. 1992. *Categorizing binary topological relations between regions, lines, and points in geographic databases*. Technical Report. 28 pages.

Ali O. Ercan, Abbas El Gamal, and Leonidas J. Guibas. 2013. Object Tracking in the Presence of Occlusions Using Multiple Cameras: A Sensor Network Approach. *ACM Trans. Sen. Netw.* 9, 2 (April 2013), 16:1–16:36. DOI:http://dx.doi.org/10.1145/2422966.2422973

Antony Galton. 2000. *Qualitative Spatial Change*. Oxford University Press, Oxford.

Torsten Hahmann and Boyan Brodaric. 2013. Kinds of Full Physical Containment. In *Spatial Information Theory (Lecture Notes in Computer Science)*. Springer, Scarborough, UK, 397–417. DOI:http://dx.doi.org/10.1007/978-3-319-01790-7_22

Ray Jackendoff. 1976. Toward an Explanatory Semantic Representation. *Linguistic Inquiry* 7, 1 (1976), 89–150. DOI:http://dx.doi.org/10.2307/4177913

Myeong-Hun Jeong and Matt Duckham. 2013. Decentralized querying of topological relations between regions monitored by a coordinate-free geosensor network. *GeoInformatica* 17, 4 (Feb. 2013), 669–696. DOI:http://dx.doi.org/10.1007/s10707-012-0174-7

Myeong-Hun Jeong, Matt Duckham, Allison Kealy, Harvey J. Miller, and Andrej Peisker. 2014. Decentralized and coordinate-free computation of critical points and surface networks in a discretized scalar field. *International Journal of Geographical Information Science* 28, 1 (Jan. 2014), 1–21. DOI:http://dx.doi.org/10.1080/13658816.2013.801578

Yu Liu, Qinghua Guo, and Maggi Kelly. 2008. A framework of region-based spatial relations for non-overlapping features and its application in object based image analysis. *ISPRS Journal of Photogrammetry and Remote Sensing* 63, 4 (July 2008), 461–475. DOI:http://dx.doi.org/10.1016/j.isprsjprs.2008.01.007

Nancy A. Lynch. 1996. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Hadi Tabatabaee Malazi, Kamran Zamanifar, Andrei Pruteanu, and Stefan Dulman. 2013. Gossip-based density estimation in dynamic heterogeneous wireless sensor networks. *International Journal of Autonomous and Adaptive Communications Systems* 7, 1-2 (Dec. 2013), 151–168. DOI:http://dx.doi.org/10.1504/IJAACS.2014.058019

Silvia Nittel. 2009. A survey of geosensor networks: Advances in dynamic environmental monitoring. *Sensors* 9, 7 (2009), 5664–5678. DOI:http://dx.doi.org/doi.org/10.3390/s90705664

Silvia Nittel, Anthony Stefanidis, Isabel F. Cruz, Max J. Egenhofer, D. Goldin, A. Howard, Alexandros Labrinidis, Samuel R. Madden, A. Voisard, and Michael F. Worboys. 2004. Report from the First Workshop on Geo Sensor Networks. *SIGMOD Rec.* 33, 1 (March 2004), 141–144. DOI:http://dx.doi.org/10.1145/974121.974146

Silvia Nittel, Niki Trigoni, Konstantinos Ferentinos, Francois Neville, Arda Nural, and Neal Pettigrew. 2007. A Drift-tolerant Model for Data Management in Ocean Sensor Networks. In *Proceedings of the 6th ACM International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE '07)*. ACM, New York, NY, USA, 49–58. DOI:http://dx.doi.org/10.1145/1254850.1254860

Gabriel Ramos-Fernández, José L. Mateos, Octavio Miramontes, Germinal Cocho, Hernn Larralde, and Bárbara Ayala-Orozco. 2004. Lévy walk patterns in the foraging movements of spider monkeys (Ateles geoffroyi). *Behavioral Ecology and Sociobiology* 55, 3 (Jan. 2004), 223–230. DOI:http://dx.doi.org/10.1007/s00265-003-0700-6

David A. Randell, Zhan Cui, and Anthony G. Cohn. 1992. Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR '92). In *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR '92)*, Vol. 92. Morgan Kaufmann Publishers, Cambridge, MA, 165–176.

Nicola Santoro. 2006. *Design and analysis of distributed algorithms*. Wiley Series on Parallel and Distributed Computing, Vol. 56. Wiley-Interscience. DOI:http://dx.doi.org/10.1002/0470072644

Carl P. L. Schultz, Hans W. Guesgen, and Robert Amor. 2006. Computer-human Interaction Issues when Integrating Qualitative Spatial Reasoning into Geographic Information Systems. In *Proceedings of the 7th ACM SIGCHI New Zealand Chapter's International Conference on Computer-human Interaction: Design Centered HCI*. ACM, New York, NY, USA, 43–51. DOI:http://dx.doi.org/10.1145/1152760.1152766

David W. Sims, Emily J. Southall, Nicolas E. Humphries, Graeme C. Hays, Corey J. A. Bradshaw, Jonathan W. Pitchford, Alex James, Mohammed Z. Ahmed, Andrew S. Brierley, Mark A. Hindell, David Morritt, Michael K. Musyl, David Righton, Emily L. C. Shepard, Victoria J. Wearmouth, Rory P. Wilson, Matthew J. Witt, and Julian D. Metcalfe. 2008. Scaling laws of marine predator search behaviour. *Nature* 451, 7182 (Feb. 2008), 1098–102. DOI:http://dx.doi.org/10.1038/nature06518

Leonard Talmy. 1983. How Language Structures Space. In *Spatial Orientation*, Herbert L. Pick Jr and Linda P. Acredolo (Eds.). Springer, 225–282. DOI:http://dx.doi.org/10.1007/978-1-4615-9325-6_11

Uri Wilensky. 1999. *NetLogo*. ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. ccl.northwestern.edu/netlogo/

Michael F. Worboys. 2012. The Maptree: A Fine-Grained Formal Representation of Space. In *Geographic Information Science (Lecture Notes in Computer Science)*, Ningchuan Xiao, Mei-Po Kwan, Michael F. Goodchild, and Shashi Shekhar (Eds.), Vol. 7478. Springer, Columbus, OH, 298–310. DOI:http://dx.doi.org/10.1007/978-3-642-33024-7_22

Michael F. Worboys. 2013. Using Maptrees to Characterize Topological Change. In *Spatial Information Theory (Lecture Notes in Computer Science)*. Springer, Scarborough, UK, 74–90. DOI:http://dx.doi.org/10.1007/978-3-319-01790-7_5

Xu Zhong, Allison Kealy, Guy Sharon, and Matt Duckham. 2015. Spatial Interpolation of Streaming Geosensor Network Data in the RISER System. In *Web and Wireless Geographical Information Systems (Lecture Notes in Computer Science)*. Springer, Grenoble, France, 161–177. DOI:http://dx.doi.org/10.1007/978-3-319-18251-3_10