GESTALT: Geospatially Enhanced Search with Terrain Augmented Location Targeting

Kent O'Sullivan†
osullik@umd.edu
University of Maryland
USA

Nicole R. Schneider†
nsch@umd.edu
University of Maryland
USA

Aleeza Rasheed
aleeza.z.rasheed@gmail.com
National Cathedral School
USA

Hanan Samet
hjs@cs.umd.edu
University of Maryland
USA

ABSTRACT

Geographic information systems (GIS) provide users with a means to efficiently search over spatial data given certain key pieces of information, like the coordinates or exact name of a location of interest. Current GIS capabilities do not enable users to search for locations using imperfect or incomplete information easily. In these cases, GIS tools help narrow down a region of interest, but users must conduct a manual last-mile search to find the exact location of interest within that region. This typically involves the user visually inspecting many remote sensing or street-view images to identify distinct landmarks or terrain features that match the partial information provided. This step of the search process is a bottleneck. Taking inspiration from the way humans recall and search for information, we present *the **G**eospatially **E**nhanced **S**earch with **T**errain **A**ugmented **L**ocation **T**argeting (**GESTALT**)*, an end-to-end pipeline for extracting geospatial data, transforming it into coherent spatial relations, storing those relations, and searching over them. We contribute a new Swan Valley Wineries dataset and a proof of concept architecture that includes multiple methods for querying spatial configurations of objects, handling uncertainty in the information known about a location or object, and accounting for the fuzzy boundaries between locations.

ACM Reference Format:
Kent O'Sullivan†, Nicole R. Schneider†, Aleeza Rasheed, and Hanan Samet. 2023. GESTALT: Geospatially Enhanced Search with Terrain Augmented Location Targeting. In *2nd ACM SIGSPATIAL International Workshop on Searching and Mining Large Collections of Geospatial Data (GeoSearch '23), November 13, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3615890.3628535

† Equal contribution by the authors.

GeoSearch '23, November 13, 2023, Hamburg, Germany

ACM ISBN 979-8-4007-0352-2/23/11.
https://doi.org/10.1145/3615890.3628535

1 INTRODUCTION

Geographic information systems (GIS) provide users with a means to efficiently search over spatial data given information like the coordinates or exact name of a location of interest. However, current GIS capabilities do not enable users to search for locations with imperfect or incomplete information easily. From psychology and neuroscience research on cognitive maps of terrain for navigation and route planning [8, 10, 18], we know that people anchor memories of a location around its visible objects and landmarks, likely hierarchically or separately relating global and local features [18]. For example, a user may remember a location by a series of visual features encountered near it, like a large building, a bus stop, and a brightly colored sign, but fail to recall its exact address or physical coordinates. In these cases, GIS tools may help narrow down the general region of interest, but the user must then perform a manual *last-mile* search to find the exact location of interest within that region. Last-mile search involves visually inspecting images to identify distinct landmarks or terrain features that match the partial information about the location. The last-mile component of the search process is a bottleneck, as it encumbers the user with the burden of sifting through many possible candidate locations until the correct one is visually identified.

Last-mile search is a common problem in the open-source intelligence community. Recent related work from *Bellingcat* highlights the need for search techniques that can accommodate queries for collections of objects in close geographic proximity to each other. Automating the last-mile search process requires addressing additional criteria, including the relative spatial configuration of objects, uncertainty in the knowledge of a location to allow for partial matches, and accounting for the fuzzy boundaries between locations, where an object may be visible from several nearby locations and can be associated with all of them to maximize the recall of the search result.

We present *GESTALT*[1,2], an end-to-end spatial search pipeline inspired by the way humans recall and search for information [7, 12, 18]. *GESTALT* extracts geospatial data, transforms it into coherent object and location relations, stores those relations, and enables search. Specifically, *GESTALT* provides the following functionality:

(1) Multiple methods for ingesting location and object tags, including automatic object detection on geotagged images.

[1] https://github.com/osullik/GESTALT_GEOSEARCH
[2] https://gestalt.umiacs.umd.edu/

(2) Density-based clustering of object tags to fuzzily assign objects to (possibly multiple) nearby locations, enabling users to ask membership questions, like "Which locations contain a swimming pool, a statue, and a palm tree?"
(3) Probabilistic search over locations based on object membership queries and with partial matches when the search constraints are too narrow and ranked results when they are too broad.
(4) A proof-of-concept instance of the *GESTALT* user interface with the Swan Valley and Washington DC regions available at https://gestalt.umiacs.umd.edu.

We contribute a new hand-labeled *Swan Valley Wineries* dataset for the last-mile spatial search problem and provide a proof of concept implementation of the proposed *GESTALT* architecture. We report the precision and recall on ground truth data and execution performance metrics on a more extensive data set to demonstrate the scalability of our search procedures.

The rest of this paper is organized as follows. In section 2, we define the *GESTALT* architecture and discuss how each subsystem contributes to our human-centric approach to automating the last-mile search. Next, in section 3, we describe the process used to generate the *Swan Valley Wineries* dataset and extract noisy object tags from geotagged images. Then, we describe the object ownership assignment process in section 4, and the search process in 5. Finally, we summarize related work in section 6 and conclude by identifying future research directions in section 7.

2 ARCHITECTURE

The architecture of *GESTALT* (Figure 1) covers four essential functions needed to perform last-mile search: data acquisition, object ownership assignment, concept mapping, and search. We describe and motivate each of them below, leaving the implementation details for sections 3 through 5.

Data Acquisition. The Data Acquisition subsystem of *GESTALT* leverages a variety of data sources to ingest or tag objects and locations, including hand-labeled data in Keyhole Markup Language [3] (KML) files, crowd-sourced object and location labels from Open Street Maps (OSM) [5], and automatically generated object tags extracted from Flickr images geolocated within the region of interest. Data Acquisition fuses heterogeneous data formats to enable common representation, storage and search.

Ownership Assignment. The Ownership Assignment subsystem of *GESTALT* maps the *objects* to the *locations* they most likely belong to using fuzzy clustering that allows objects to be assigned to multiple locations. *GESTALT*'s assignment protocol accounts for uncertainty in geotag coordinates of locations and objects and allows for the possibility that objects can be visible from multiple locations in the real world.

Data Structures. *GESTALT* creates several data structures (inverted index, location-object concept map, object-object concept map) to facilitate spatial search over objects and locations. The *inverted index* maps each object class to the set of locations that contain objects of that type, enabling membership queries, which prune the search space for downstream spatial searching. The *location-object structure* treats the coordinates of the *location* as the division point on the north-south and west-east axes and maps the objects for each location to the corresponding quadrant they lie in with respect to the location. The *object-object concept map* (for each location) encodes the implicit directional relationships between the objects at that location using sparse matrices that can be recursively searched to eliminate candidate locations that do not match the spatial query specification.

Search. The Search subsystem of *GESTALT* enables the user to identify locations of interest based on partial knowledge about the existence and geospatial arrangement of objects at that location. The search process is designed for the way humans naturally conceptualize and describe locations and directions - by drawing a map. Queries can be issued using a pictorial specification, relating objects through relative cardinal relations (like Northwest, etc.) or cardinality-invariant relations (when the user does not recall any cardinal information). The Search subsystem supports fuzzy searching to return inexact matches when query terms are too narrow and ranking that reflects the uncertainty in an object's identification, tagging, and assignment to a location.

3 DATA ACQUISITION

GESTALT enables last-mile search by encoding visual and geospatial data hierarchically within a given *region*, using two types of geotags: *object* tags and *location* tags.

Regions represent a limited physical area of interest within which a last-mile search is performed. Regions can be defined arbitrarily and represent an administrative boundary like a city, suburb, or general geographic area. We define regions by bounding boxes for compatibility with the OSM and Flickr APIs.

Objects represent any physical entity within the region of interest. For example, a *tree, building, lake, bridge, gate* or *sign* could be an object. *Objects* can also have attributes that provide amplifying information about them, like *color, material, size, etc.*.

Locations represent physical entities that *do* something, giving them a purpose beyond that of objects. Locations can contain meaningful groupings of objects determined by ownership, proximity, or utility. Examples of locations include *businesses, attractions, properties, etc.*. *Locations* have *objects* associated with them, and *GESTALT* enables users to query for locations given a partial set of knowledge about the objects at those locations.

3.1 Location and Object Tags

To maximize dataset coverage and demonstrate the flexibility of *GESTALT*, we support three methods of ingesting objects and two methods of ingesting locations:

(1) Ingesting KML Files that contain manually annotated *objects* or *locations* and their coordinates.
(2) Querying the OSM API to ingest crowd-sourced *object* or *location* tags and their coordinates.
(3) Automatically detecting *objects* in geolocated photos pulled from the Flickr API.

Upon ingest, each object or location is assigned a confidence score, reflecting the certainty that it exists at the coordinates tagged and is of the type annotated. For results reported in this paper, we adopt the rule that hand-labeled objects and locations receive a confidence score of 1.0, OSM-labeled objects and locations receive

[3] https://developers.google.com/kml/documentation/kml_tut

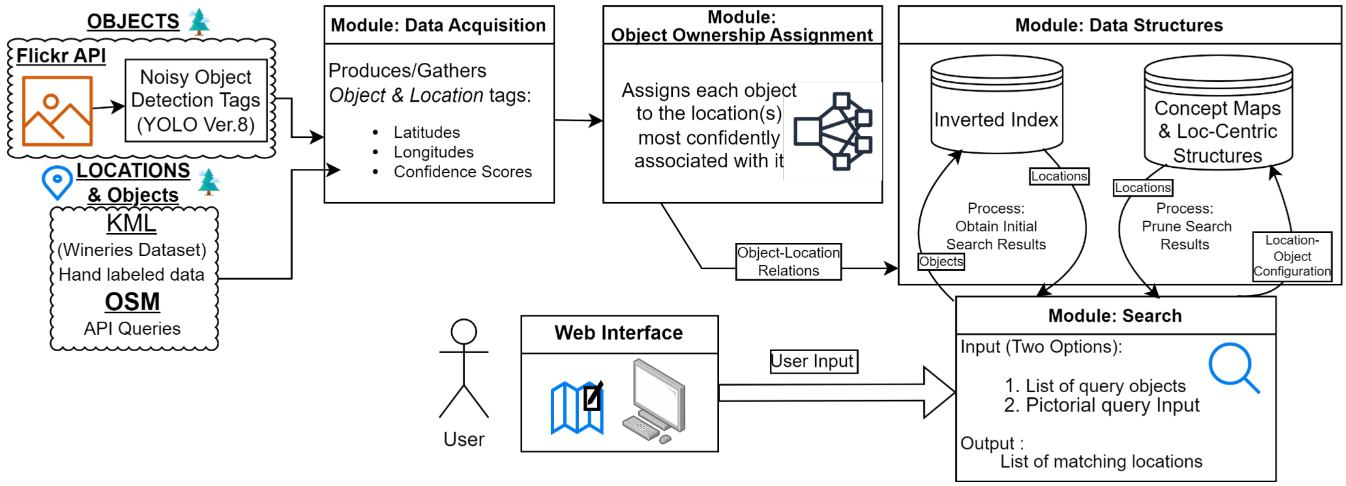


Figure 1: *GESTALT* consists of a data collection module, an ownership assignment module, a concept mapping module, and a search module.

a score of 0.75, and automatically-detected objects receive the confidence score reported by the underlying object detection model. Table 1 summarizes the objects and locations currently in *GESTALT*.

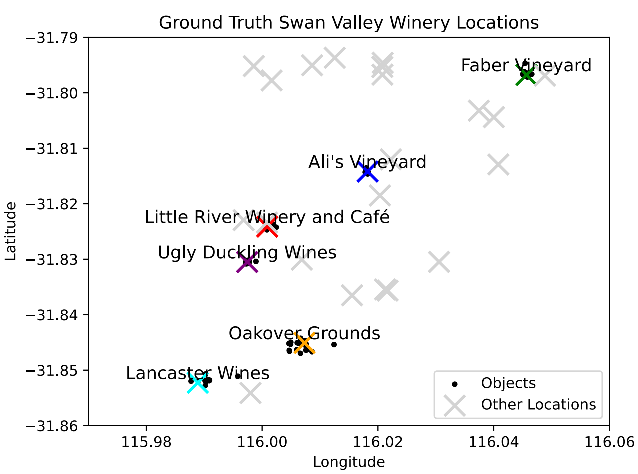


Figure 2: The Swan Valley Wineries dataset has 6 locations with annotated objects and another 30 without manually annotated objects but with confirmed location coordinates.

3.2 Ingest Methods

3.2.1 Ground Truth Hand Labeled Tags. We support ingesting hand-labeled objects and locations manually annotated by a trustworthy source. For benchmarking purposes, we curated the *Swan Valley Wineries* dataset containing 31 ground truth location tags for wineries (and five for breweries) in the Swan Valley Region of Western Australia and 146 ground truth object tags associated with 6 of those wineries. The tags consist of an object name, its latitude & longitude, and any descriptive markings written as key:value pairs. We annotated the objects manually using *Google Earth Professional version 7.3*. The data sources include on-the-ground knowledge, manual inspection of satellite imagery, street-view imagery, and publicly available aerial photos. The objects tagged are representative, not exhaustive. Attributes of the objects (e.g., color, size, material) are also recorded. The object tags are aligned with their corresponding winery location in the dataset.

3.2.2 Open Street Maps Tags. We also support ingesting object and location tags by querying the OSM API for crowd-sourced Geodata, including *Nodes* (point data for objects and locations), *ways* (line data for roads, creeks, railways, etc.), and *relations* (for relationships between elements, like suburb-state). While businesses, attractions, and other *locations* are commonly annotated by the OSM community, *objects* are more sparsely tagged as they are typically less interesting subject matter. To ingest objects, *GESTALT* pulls all nodes within a given bounding box that have at least one tag (like "name," "craft," etc.) and prunes them to remove results not likely referring to objects or that lack detail to be resolved. To ingest locations, *GESTALT* uses a similar process to objects but excludes results that lack features like names, street addresses, and phone numbers.

3.2.3 Noisy Image-based Tags. The third and most important method by which *GESTALT* ingests objects is automatic object detection. We query the Flickr API for images uploaded within a given region (bounding box) since January 1^{st} 2020. Given those images and their EXIF metadata, the Object Detection module uses pre-trained YOLO v.8 [4] to identify objects in each image from 80 classes (based on the COCO dataset [5]). *GESTALT* stores those objects with their coordinates determined by the geolocation of the image and their confidence score determined by YOLO's confidence at the detection

[4] https://github.com/ultralytics/ultralytics
[5] https://cocodataset.org/

step. For the Swan Valley Region, we retrieved 462 images, and for the Washington DC Region, we retrieved 4,000 images.

Region	Type	Source	Label Method	# Tags
Swan Valley[6]	Objects	KML file	Hand Labeled	146
	Objects	OSM B-Box	Crowd-Sourced	2466
	Objects	Flickr B-Box	Object detection	1893
	Locations	KML file	Hand Labeled	31
	Locations	OSM B-Box	Crowd-Sourced	308
Washington D.C.[7]	Objects	OSM B-Box	Crowd-Sourced	60123
	Objects	Flickr B-Box	Object detection	31065
	Locations	OSM B-Box	Crowd-Sourced	12179

Table 1: Location and object datasets in *GESTALT*. Swan Valley region data with hand labels are part of the Swan Valley Wineries dataset we contribute. A *Combined* dataset including all labeling methods for the Swan Valley region is used for some testing.

4 OWNERSHIP ASSIGNMENT

Ownership assignment enables search over locations using queries about the objects contained at or near those locations.

4.1 Object Ownership Problem Definition

We formulate the last-mile search problem in terms of *locations*, which are searchable entities that 'own' or contain any number of *objects*. To support user queries, *GESTALT* must associate objects with their parent locations. We term this problem *Object Ownership Assignment* and define it as follows. Given a collection of locations and objects within a region, *Ownership Assignment* seeks to correctly assign each object to its 'parent' location. Ownership Assignment amounts to a clustering problem where we assign points (objects) to centroids (locations) that are determined apriori. The objects are not uniformly distributed across locations; some locations may not have associated tagged objects.

Humans do not see the world in regular grid lines like on a map. Hence, the ownership assignment process is naturally inexact, and objects are 'shared' between locations where appropriate. For example, a winery may have a shed nearby that is visible both from that winery and a neighboring one. In this case, the object can be helpful when searching for either location. We address this aspect of the problem by modifying our method to allow for fuzzy object-location assignment, effectively increasing the recall.

4.2 Object Ownership Assignment Dataset

We evaluate the Ownership Assignment task on two datasets: the *hand-labeled* Swan Valley Wineries dataset and a *Combined* dataset that includes all of the object and location tags from the Swan Valley Wineries dataset in addition to the noisy object tags ingested from Flickr and OSM. Since we have ground truth labels for the winery locations and their objects in the Swan Valley Wineries dataset, we report both precision and recall on this dataset. The Combined dataset presents a more realistic and challenging scenario for Ownership Assignment than the Swan Valley Wineries dataset alone, given its more extensive set of locations and a large set of object tags to assign to those locations. However, since the Combined dataset includes noisy tags for which we have no ground truth labels, we do not report precision. Instead, we measure the recall of the same set of ground-truth queries in the presence of the additional noise. Table 2 contains the recall (and precision, where applicable) on those locations and objects.

[6]BoundingBox:['115.96168231510637', '-31.90009882641578', '116.05029961853784', '-31.77307863942101']
[7]BoundingBox:['-77.120248', '38.791086', '-76.911012', '38.995732']

4.3 Object Ownership Assignment Method

Our method for Object Ownership Assignment is outlined in Algorithm 1. We use DBSCAN [3] to cluster the objects on the first pass and prune out noisy objects that are unlikely to be associated with any particular location. After clustering the objects, we determine the centroids of the relevant object clusters, calculate the confidence scores in the object-cluster assignments, and then map the centroids to the nearest known location tags using a nearest-neighbor search on a KD-Tree Objects are then added to nearby clusters using an adjustable 'fuzziness' parameter. When calculating the confidence scores, we normalize the object-centroid distances (omitting objects in the null cluster, which has no meaningful centroid and would skew the normalization). This confidence score (between 0 and 1) measures how far a given object is from its cluster's centroid, assigning higher scores to objects near the centroid and lower ones to objects far from the centroid. We take this approach rather than using a static threshold parameter (like within x distance) to account for the variety in object density of the region under search. We adopt the convention of assigning null cluster objects a confidence score of 0.5 since these objects are deemed noise that is not relevant for finding locations. To further account for uncertainty in object tags, we add an adjustable parameter c to the method, which allows for a varying degree of *fuzzy assignment* of objects to clusters. By increasing the parameter value, we can allow for objects to be assigned to multiple clusters if they are close to multiple centroids (within c% of the largest object-centroid distance in the dataset, the same value used to normalize the confidence scores).

Algorithm 1 Object to Location Ownership Assignment Algorithm

```
Locs a list of locations and their tagged coordinates
Objs a list of objects and their coordinates
- - - - - - - - - -
procedure ObjectOwnershipAssignment(Locs, Objs)
    Clusters ← DBSCAN(Objs)
    for Each Cluster in Clusters except NULL cluster do
        C ← Calculate centroid of Cluster
        for Obj in Cluster do
            Obj.prob ← 1 − normalize(dist(Obj, C))
        end for
    end for
    for Each Obj in NULL Cluster do
        Obj.prob ← 0.5
    end for
    for Each Cluster in Clusters Except NULL Cluster do
        Cluster.Loc ← Closest location to C in Locs
    end for
end procedure
```

Dataset	Method	Fuzzy Param	Precision	Recall
Swan Valley Wineries	Exact	$c = 0$	1.0	0.89
	Fuzzy	$c = 2$	0.85	0.89
	Fuzzy	$c = 4$	0.85	0.89
	Fuzzy	$c = 6$	0.85	0.89
	Fuzzy	$c = 8$	0.85	0.89
	Fuzzy	$c = 10$	0.85	0.89
Combined	Exact	$c = 0$	-	0.88
	Fuzzy	$c = 2$	-	0.88
	Fuzzy	$c = 4$	-	0.91
	Fuzzy	$c = 6$	-	0.94
	Fuzzy	$c = 8$	-	0.97
	Fuzzy	$c = 10$	-	0.98

Table 2: Object-to-location assignment results for Swan Valley Wineries and Combined datasets. Increasing fuzziness improves recall on the noisier *Combined* dataset.

We run all Swan Valley Wineries experiments with DBSCAN parameter $\epsilon = \frac{0.05}{6371}$ (~50m) and DC experiments with $\epsilon = \frac{0.01}{6371}$ (~10m). For both, the MinClusterSize=3. We choose these values using the knowledge that the environment of interest (Swan Valley region) is semi-rural and more likely to be spread out than the urban Washington D.C. The minClusterSize is not critical to the assignment process because fuzzy assignment allows objects to be added to nearby clusters, which overcomes too fine a clustering outcome.

4.4 Object Ownership Assignment Results

We show the Object-to-location assignment results for the Swan Valley Wineries and the *Combined* datasets in Table 2. Increasing the fuzziness parameter c increases the chance that objects are assigned to multiple clusters, which improves recall on the noisier *Combined* dataset. As expected, increasing the likelihood that objects are assigned to multiple clusters in the less noisy Swan Valley Wineries dataset reduces the precision but, interestingly, does not show the benefit in recall that it did for the noisier *Combined* dataset.

Overall, the object ownership results show that our method handles the noisy object tags well, especially on the larger dataset. By using DBSCAN for the initial clustering, we observe a noise reduction effect as objects that do not belong to any locations are assigned to the NULL cluster (i.e., a mistagged singular object with nothing else around it for miles in any direction). A similar recall on the ground-truth labels is achieved despite adding a considerable quantity of noisy object tags (in the *Combined* dataset).

The downside of using DBSCAN in this context is that it determines the centroids of the object clusters based on the object density, and then we must map those to our known location centroids in a post-processing step. Ideally, a better solution would start with the centroids and cluster around them while retaining the noise reduction effects of DBSCAN. We leave a detailed comparison of Object Ownership Assignment techniques as an interesting avenue of future study for the last-mile search problem.

5 SEARCH

The core function of *GESTALT* is to perform last-mile search given partial or uncertain information. The user is assumed to know the general region of interest and some information about the objects at the location they seek. Under these conditions, the search problem can be framed in several ways, which we describe in subsection 5.2 in increasing order of complexity and utility.

5.1 Search Query Sets

We tested *GESTALT*'s search performance by creating two sets of queries (*Swan Valley ground truth queries*, and *D.C. worst-case queries*). For reproducibility, we provide both sets of queries in the form of object names and their spatial configurations. We also provide the expected results set for each Swan Valley query.

The *Swan Valley ground truth queries* are designed to enable comparison of precision and recall of different spatial search methods. To construct them, we hand-labeled the ground truth responses for 58 spatial queries on object configurations for arbitrary objects belonging to 6 wineries in the Swan Valley Wineries dataset.

The *D.C. worst-case queries* are designed to test *GESTALT*'s performance under the worst-case search conditions where the user recalls no highly distinctive features of their target location. We construct these queries on the noisy Washington D.C. dataset, using the ten most common object classes as query terms (e.g., crossing, traffic signals, street lamp, bus stop).

Algorithm 2 Membership Search

Q a list of query terms to search for
II an inverted index with objects as keys and locations as values
\- - - - -
procedure SEARCH(Q, II)
  $SearchResult \leftarrow []$
  for Each P in Q **do**
    Retrieve set of Locations $II[P]$ and add to $SearchResult$
  end for
  return intersection of sets of Locations in $SearchResult$
end procedure

Algorithm 3 Ranked Membership Search

Q a list of query terms to search for
II an inverted index with objects as keys and locations as values
SearchResult Set of candidate locations returned from **SEARCH()**
\- - - - -
procedure RANK($SearchResult$, Q, II)
  $RankedSearchResult \leftarrow$ Empty Ordered Dictionary
  for Each location Loc in $SearchResult$ **do**
    $prob \leftarrow 1$
    for Each query point P in Q **do**
      $prob \leftarrow prob \times P.prob$
    end for
    $RankedSearchResult[Loc] \leftarrow prob$
  end for
  $RankedSearchResult \leftarrow$ Sort $RankedSearchResult$ by values
  return $RankedSearchResult$
end procedure

5.2 Search Method

5.2.1 ***Exact membership search.*** The simplest search function in *GESTALT* (Algorithm 2) takes a set of query terms representing objects the user knows are at a location and performs the appropriate look-ups and set intersections to determine which locations are a match (contain *all* those objects).

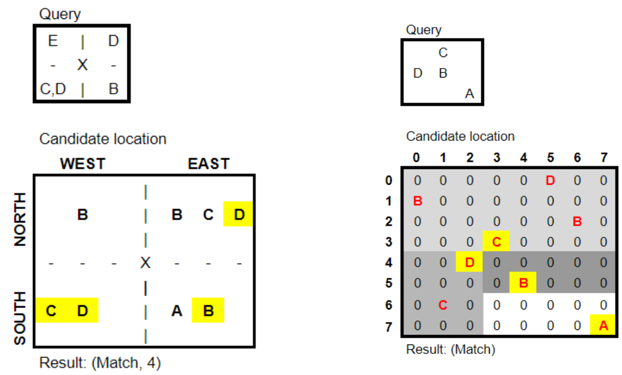
5.2.2 ***Ranked membership search.*** When the exact membership search returns a large number of matches, such as for a broad query (i.e., locations that have a tree and a bench), ranking those locations can help narrow the results. Using Algorithm 3, we aggregate the confidence scores from the object tagging and ownership assignment stages to determine the overall likelihood that a given location contains the object of interest. These scores are then aggregated per location for the relevant query objects, and the final scores determine the ranking of the results.

5.2.3 ***Fuzzy Membership Search.*** When the exact membership search returns no matches, we use a fuzzy search procedure (Algorithm 4) to find a set of partial matches based on the most discriminative object(s) in the list of query terms. The most discriminative terms are emphasized since rare objects are memorable and uniquely identify locations more than common objects.

Algorithm 4 Fuzzy Membership Search

    Q *a list of query terms to search for*
2: ***II*** *an inverted index with objects as keys and locations as values*
    - - - - -
4: **procedure** FUZZYSEARCH(Q,II)
        SearchResult ← **II.SEARCH**(Q)
6:     **if** *SearchResult* is Empty **then**
            P ← Pop most discriminative term from Q
8:         *SearchResult* ← II.**SEARCH**(Q)
            if *SearchResult* is not Empty **then**
10:                Skip to Line 6
            else
12:                *SearchResult* ← **II.SEARCH**(Q.remove(P)
            end if
14:     **end if**
        if *SearchResult* has more than 1 item **then**
16:         **return** II.**RANK(***SearchResult*, Q**)**
        end if
18: **end procedure**

5.2.4 ***Spatial Search.*** *GESTALT*'s core search functionality comes from the spatial search methods it enables, which rely on the spatial data structures and algorithms described in *COMPASS* [13]. The location-centric search method (Location-Object search) allows the user to query for locations based on the relative position of objects in each of the four cardinal directions, depicted in Figure 3a. The object-centric search method (Object-Object search) allows the user to specify the relative layout of objects relative to each other, regardless of where they are relative to the location under search (Figure 3b. A third form of spatial search (Cardinality Invariant Object-Object search) is also supported, which mimics the Object-Object search but removes the requirement that the query pattern be provided with the correct cardinal alignment compared to the actual pattern of objects in the real world.



(a) Location-object search. **(b) Object-object search.**

Figure 3: Spatial search by cardinal direction to location sought (Figure 3a) and by directional relationship between objects (Figure 3b).

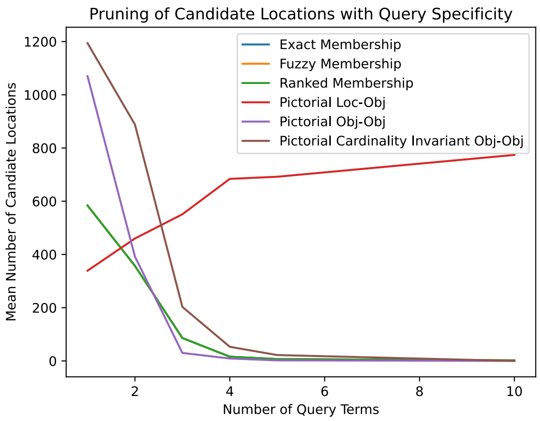
5.3 Search Results

Search Method	Metric	Results
Location-Object Search	Mean Precision	1.000
	Mean Recall	0.800
Object-Object Search	Mean Precision	0.947
	Mean Recall	0.737
Object-Object Cardinality Invariant Search	Mean Precision	0.825
	Mean Recall	0.816

Table 3: Spatial search performance results across 58 ground-truth pictorial queries run on the *Combined Swan Valley Wineries* dataset.

Search Performance. To evaluate *GESTALT*'s search process from end to end in a noisy environment, we test it using the *Swan Valley ground truth query set* and the *Combined* dataset. The results (Table 3) show that *GESTALT* has a high precision and recall on all three methods of spatial search.

The cardinality-invariant version of the Object-Object search outperforms the non-invariant version in recall at the cost of precision. The variance in performance is because rotated versions of the object configuration in some ground truth queries may happen to exist at other locations in the noisy *Combined* dataset despite not being labeled in the ground truth, causing false positives. In general, we observed that increasing the number of objects in the query reduced the false positive rate since additional constraints led to fewer random collisions with other locations not considered during the hand-labeling. The queries tested contained between one and four objects per query, sufficient to observe reasonable precision overall, indicating the discriminative power of spatial configurations of objects as a search constraint.



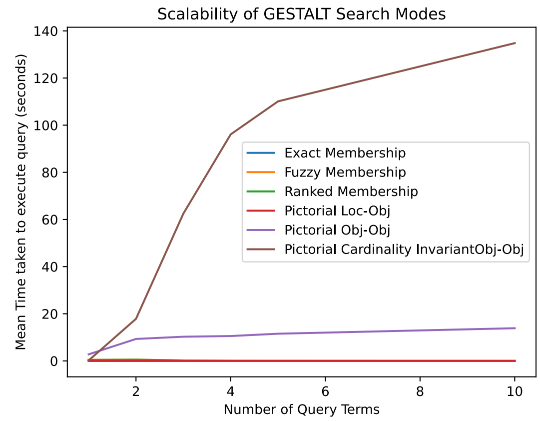


Figure 4: Spatial search complexity results measured in the number of candidate locations and query response times for each type of spatial query *GESTALT* supports. Measured on the Washington D.C. Dataset (12,179 Locations, 91,188 objects).

Search Complexity. A detailed theoretical analysis of search complexity for the three forms of spatial search is presented in *COMPASS* [13]. To assess the scalability of *GESTALT* on real data, we measure query response times on the *D.C. worst-case queries* and the *Washington D.C. dataset* (which contains over 91,000 objects and 12,000 locations). Figure 4 shows these results as the number of query terms increases. For the three membership searches (exact, ranked, and fuzzy), the response times decrease as the number of objects specified in the query increases (i.e., as the pool of possible locations meeting the query specification narrows), following the trend in Figure 4 which shows the aggressive pruning that occurs as the queries become more specific. Critically, this same effect is achieved for the object-object spatial searches, where the recursive pruning of the search space quickly eliminates any candidates that are not viable matches to the pictorial query specification. The location-object search does not show this effect because it counts the number of objects matching the query configuration and uses that to rank candidates, so no early stopping can be done.

6 RELATED WORK

GESTALT takes a human-centric approach to geospatial search, borrowing ideas from human spatial cognition, navigation, and spatial reasoning. We address related work in human spatial reasoning and pictorial querying and then compare *GESTALT* with the current closest geospatial search tool.

6.1 Human Spatial Cognition and Reasoning

People tend to anchor memories on objects they see while experiencing a location [7] and use them to find it again. Even robots we design to do perception need visual anchoring [12]. Reliance on anchoring underpins *GESTALT*'s human-centric approach to last-mile spatial search wherein the goal is to retrieve locations in a region given information about their objects. Last-mile search forms the reciprocal problem to the online game *Geoguessr* [8], which challenges users to determine the broad region of a photograph based on the visible objects. Successful players of *GeoGuessr* show that enough objects, even common ones, have the power to substantially prune a large search space down to just a few, or even one candidate region [1]. *GESTALT*'s approach to geospatial search closely follows the *Winthrop Method* [8] which was developed in Northern Ireland in the 1970s to detect clandestine weapons caches. The core principle behind this method is that humans form mental models of terrain based on objects in the environment and then use those to navigate. Other research has shown that when revisiting locations, the objects can even serve as keys to other memories of that location [10]. Specific to geospatial search systems, Schwering et al. synthesize the findings of multiple user studies into geospatial reasoning errors, highlighting that most humans are much better at recalling relative object positions than cardinal directions or distances between objects [14].

6.2 Pictorial and Spatial Pattern Matching

The spatial search functionality of *GESTALT* leverages a pictorial query interface to bridge the gap between the user's mental model of objects and the geospatial data available to search. Pictorial query interfaces have previously allowed users to place objects and assign constraints in a *drag-and-drop* [2, 4, 16, 17], or *query-by-sketch* [14] manner. Each has its merits, but approaches that allow users to specify objects' relative positions are better suited for the *GESTALT* search paradigm. Regardless of the interface, most query systems use an underlying query engine for *spatial pattern matching* driven by *set-intersection*, *qualitative spatial reasoning/constraint satisfaction*, or *subgraph matching* [13]. None of these three approaches have proven to sufficiently balance scalability with result quality at a production system scale. *GESTALT* addresses the complexity issues by successively pruning the search space, beginning with the last-mile search region, pruning out any locations that fail to contain the query terms, and then commencing the spatial search, which involves recursively pruning the search space even further, until a candidate location is determined to match or not match the configuration specified in the pictorial query.

[8] https://www.geoguessr.com/

6.3 Automated Geospatial Search

In recent months, the Open Source Intelligence platform *Bellingcat* released *Bellingcat OSM Search* [19], a tool also aimed at pruning search space using tagged objects. Using drop-downs and sliding bars, a user can specify a text-based query for OSM-tagged objects that appear within an adjustable distance from each other inside a region of interest. Although aimed at addressing the same last-mile search problem as *GESTALT*, there are several key limitations to the Bellingcat OSM Search Tool that *GESTALT* overcomes.

(1) Their tool only accounts for the distance between objects, not geographic configuration. *GESTALT* allows users to leverage the information they remember about how the objects relate to each other spatially to improve the search result.
(2) Their tool is limited to data tagged in OSM, which is sparse for the object tags *GESTALT* leverages for pruning.
(3) Their tool does not rank results, which is critical when object tags are noisy, and users are uncertain about the partial information they recall.
(4) Their tool returns 0 results if a query fails to match on every term in the region or window of interest, which is problematic when dealing with noisy, incomplete data where the likelihood of any given object being tagged is low. *GESTALT* uses a fuzzy matching procedure to return a partial match based on the most discriminative query term when the entire query yields no results, assuming that a partial answer is better than no answer.

7 CONCLUSION

The last-mile search problem involves finding locations matching some set of spatial constraints that are not easily query-able with a GIS tool. This typically includes visual landmarks or terrain features that comprise a partial or uncertain set of information about the location of interest. Solving the last-mile search problem without manual inspection of images requires spatial search techniques beyond just searching by geographic proximity (i.e. nearest neighbor searching). Inspired by the way humans recall and search for information, we developed *GESTALT*, a pipeline that (to our knowledge) is the first to enable querying for locations based on the spatial configuration of nearby objects given partial information about them. We automatically detect objects from geotagged images, fuzzily assign objects to (possibly multiple) locations, enable pictorial querying on spatial relations between objects, and perform probabilistic search over locations, returning partial matches when the search constraints are too narrow, and ranked results when they are too broad. *GESTALT* shows high recall on the ground truth benchmark dataset we contribute and easily scales to the larger, noisier datasets we tested it on. This work invites new avenues of research in improving human-centric spatial search. Advances in computer vision can be leveraged to adjust spatial search for uncertainty in object position within an image scene using the camera's bearing information [6, 9, 11, 15]. Techniques can also be developed to handle querying across additional dimensions, like time, object attribute values, and object size, scale, and quantity.

ACKNOWLEDGMENTS

This work was sponsored in part by the NSF under Grants IIS-18-16889, IIS-20-41415, and IIS-21-14451, and the Australian-American Fulbright Commission. Its contents do not necessarily represent the official views of the Fulbright Program.

REFERENCES

[1] Berners-Lee B. . 2023. The semiotics of digital cartography at the Geoguessr interface: A practice-oriented case study. *New Media & Society* (2023), 14614448231160132.
[2] Di Loreto F. , F. Ferri, F. Massari, and M. Rafanelli. 1996. A pictorial query language for geographical databases. In *Proceedings of the workshop on Advanced visual interfaces*. 233–244.
[3] Ester M. , H.-P. Kriegel, J. Sander, and X. Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (Portland, Oregon) *(KDD'96)*. AAAI Press, 226–231.
[4] Folkers A. , H. Samet, and A. Soffer. 2000. Processing pictorial queries with multiple instances using isomorphic subgraphs. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, Vol. 4. IEEE, 51–54.
[5] Haklay M. and P. Weber. 2008. Openstreetmap: User-generated street maps. *IEEE Pervasive computing* 7, 4 (2008), 12–18.
[6] Hays J. and A. A. Efros. 2008. IM2GPS: estimating geographic information from a single image. In *2008 ieee conference on computer vision and pattern recognition*. IEEE, 1–8.
[7] Helbing J. , D. Draschkow, and M. L.-H. Võ. 2020. Search superiority: Goal-directed attentional allocation creates more reliable incidental identity and location memory than explicit encoding in naturalistic virtual environments. *Cognition* 196 (2020), 104147. https://doi.org/10.1016/j.cognition.2019.104147
[8] Keatley D. , C. O'Donnell, B. Chapman, and D. D. Clarke. 2021. The psycho-criminology of burial sites: developing the winthropping method for locating clandestine burial sites. *Journal of Police and Criminal Psychology* (2021), 1–10.
[9] Liu Y. , J. Jiang, J. Sun, L. Bai, and Q. Wang. 2020. A survey of depth estimation based on computer vision. In *2020 IEEE Fifth international conference on data science in cyberspace (DSC)*. IEEE, 135–141.
[10] Miller J. F. , M. Neufang, A. Solway, A. Brandt, M. Trippel, I. Mader, S. Hefft, M. Merkow, S. M. Polyn, J. Jacobs, et al. 2013. Neural activity in human hippocampal formation reveals the spatial context of retrieved memories. *Science* 342, 6162 (2013), 1111–1114.
[11] Ming Y. , X. Meng, C. Fan, and H. Yu. 2021. Deep learning for monocular depth estimation: A review. *Neurocomputing* 438 (2021), 14–33.
[12] Oliveira M. , L. Seabra Lopes, G. H. Lim, S. H. Kasaei, A. M. Tomé, and A. Chauhan. 2016. 3D object perception and perceptual learning in the RACE project. *Robotics and Autonomous Systems* 75 (2016), 614–626. https://doi.org/10.1016/j.robot.2015.09.019
[13] O'Sullivan K. , N. R. Schneider, and H. Samet. 2023. COMPASS: Cardinal Orientation Manipulation and Pattern-Aware Spatial Search. *Under Review at GeoSearch 2023* (2023).
[14] Schwering A. , J. Wang, M. Chipofya, S. Jan, R. Li, and K. Broelemann. 2014. SketchMapia: Qualitative representations for the alignment of sketch and metric maps. *Spatial cognition & computation* 14, 3 (2014), 220–254.
[15] Snavely N. . 2011. Scene reconstruction and visualization from internet photo collections: A survey. *IPSJ Transactions on Computer Vision and Applications* 3 (2011), 44–66.
[16] Soffer A. and H. Samet. 1997. Pictorial query specification for browsing through image databasess. In *Proceedings of the Second International Conference on Visual Information Systems*. 117–124.
[17] Soffer A. and H. Samet. 1998. Pictorial query specification for browsing through spatially referenced image databases. *Journal of Visual Languages & Computing* 9, 6 (1998), 567–596.
[18] Weisberg S. M. and N. S. Newcombe. 2016. How do (some) people make a cognitive map? Routes, places, and working memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 42, 5 (2016), 768.
[19] Williams L. . 2023. Finding Geolocation Leads with Bellingcat's OpenStreetMap Search Tool. *Bellingcat Blog* (May 2023).