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
USA

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

† Equal contribution by the authors.

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. However, current GIS capabilities do not enable users to easily search for locations about which they have imperfect or incomplete information. In these cases, GIS tools may help with narrowing down to the general region of interest, but a manual last-mile search must then be performed by the user to find the exact location of interest within that region, which typically involves the visual inspection of remote sensing images or street-view images to identify distinct landmarks or terrain features that match the partial information known about the location. This step 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. 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**)*[1], an end-to-end pipeline for extracting geospatial data, transforming it into coherent object-location relations, storing those relations, and searching over them. We contribute a new gold standard 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 *Proceedings of ACM Conference (GEOSEARCH'23).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

[1]https://gestalt.umiacs.umd.edu/

GEOSEARCH'23, November 2023, Hamburg, Germany

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

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 easily search for locations about which they have imperfect or incomplete information. From psychology and neuroscience research on cognitive maps of terrain for navigation and route planning [7, 9, 20], we know that people anchor memories of a location around its visible objects and landmarks, likely hierarchically, or separately relating global and local features [20]. 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 typically 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, where analysts must identify a specific location within a general region using incomplete or uncertain information. The process is a data fusion and search task, where information from reports is manually compared to the visual features of remote sensing images and street-view photographs until the expected configuration of objects is found, indicating the location is a match. Recent related work from *Bellingcat* [2] 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 several additional criteria, including accounting for the spatial configuration of objects concerning each other, handling uncertainty in the information known about a location by relaxing the search constraints until a partial match is found, and accounting for the fuzzy boundaries between locations, whereby an object may be visible from several nearby

[2]https://www.bellingcat.com/resources/how-tos/2023/05/08/finding-geolocation-leads-with-bellingcats-openstreetmap-search-tool/

locations and can be associated with all of them to maximize the recall of the search result.

We present *GESTALT*[3], an end-to-end spatial search pipeline inspired by the way humans recall and search for information [6, 12, 20]. *GESTALT* extracts geospatial data, transforms it into coherent object and location relations, stores those relations, and enables querying over them. Specifically, *GESTALT* provides the following functionality:

(1) Multiple methods for ingesting location and object tags, including automatically using object detection methods on geotagged images.
(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) Performing probabilistic search over locations based on object membership queries and returning 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 datasets 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 the ground truth data and execution performance metrics on a more extensive set of data to demonstrate the scalability of the 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 gold standard wineries dataset and extract noisy object tags from geotagged images. Then, we describe the object ownership assignment process in section 4, and the search problem 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 KML files, crowd-sourced object and location labels from OSM, and automatically generated object tags it extracts from Flickr images geolocated within the region of interest. Data Acquisition focuses on fusing data in various formats from various sources into a standard format so the data can be transformed and searched.

Ownership Assignment. The Ownership Assignment subsystem of *GESTALT* maps the *objects* to the *locations* they most likely belong to using a fuzzy clustering that allows objects to be assigned to multiple locations. We use DBSCAN [2] to cluster the objects on the first pass and prune out noisy objects that are unlikely to be associated with any particular location. The resulting cluster centroids are then mapped to locations based on proximity using a nearest-neighbor search on a KD-Tree, and objects are added to nearby clusters using an adjustable 'fuzziness' parameter. *GESTALT*'s assignment protocol accounts for uncertainty in tag 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 enable successive pruning for 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 collection 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 narrowly defined and ranking to reflect the uncertainty in any given object's identification, tagging, and assignment to a location.

3 DATA ACQUISITION

GESTALT enables last-mile search by encoding visual and geospatial data 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 will be performed. Regions can be arbitrary and represent an administrative boundary like a city, suburb, or general geographic area. Regions are defined by bounding boxes for compatibility with the OSM and Flickr APIs.

Objects represent any physical entity located 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, including things like *color, material, size, species, 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

[3]https://github.com/osullik/GESTALT_GEOSEARCH

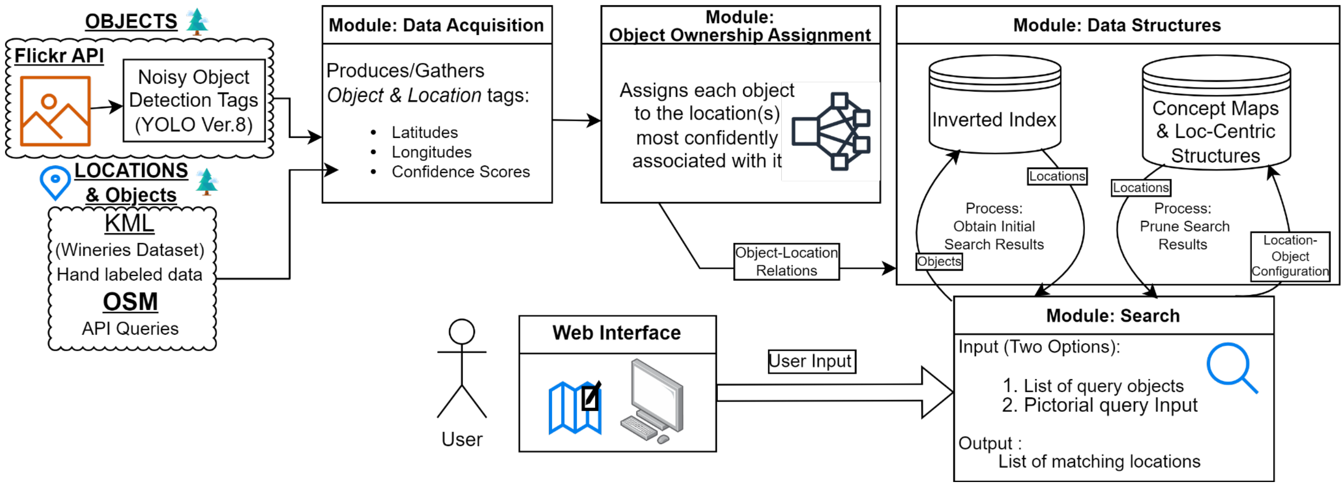


Figure 1: *GESTALT* consists of a data collection module, an ownership assignment module, a concept mapping module, and a search module.

GESTALT enables users to query for locations given a partial set of knowledge about the objects at those locations.

3.1 Object Tags

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

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

Upon ingest, each object is assigned a confidence score, reflecting the certainty that the object tagged at those coordinates exists and is of the type annotated. For results reported in this paper, we adopt the rule that hand-labeled objects receive a confidence score of 1.0, for demonstration purposes OSM objects receive a score of 0.75, and objects labeled by the object detector receive the confidence score reported by the object detection model. Table 1 contains a summary of the objects currently in *GESTALT*.

3.2 Location Tags

We support two methods of ingesting locations:

(1) Ingesting KML Files that contain manually annotated locations and their coordinates.
(2) Querying the Open Street Maps (OSM) API to ingest crowd-sourced location tags and their coordinates.

Similar to the approach with object tags, we assign a confidence to each location tag ingested based on the source providing it.

3.3 Ingest Methods

3.3.1 Ground Truth Hand Labeled Tags. Hand-labeled objects and locations have been manually annotated by a trustworthy source and are assumed to be correctly labeled and geotagged. 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 six of those wineries.

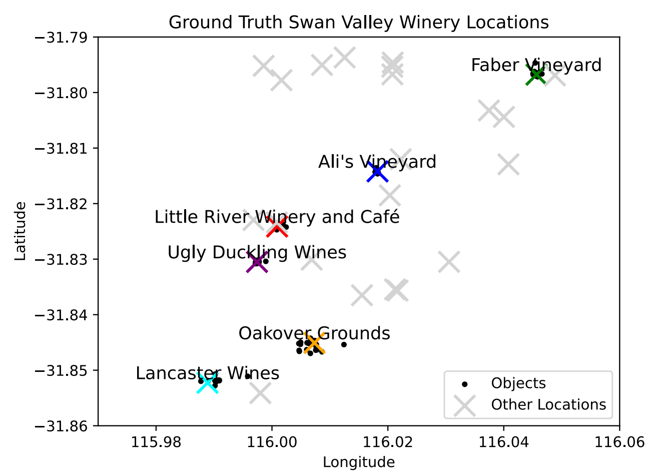


Figure 2: The Swan Valley Wineries dataset has six locations with annotated objects and another 30 without manually annotated objects but with confirmed location coordinates.

The wineries dataset tags are stored in Keyhole Markup Language [4] (KML). The tags consist of an object name, its latitude & longitude, and any descriptive markings written as key:value pairs.

The object tagging was conducted manually by a single annotator using *Google Earth Professional* [5] *version 7.3*. The data sources include on-the-ground knowledge, manual inspection of satellite imagery, street-view imagery, and publicly available area photos.

[4] https://developers.google.com/kml/documentation/kml_tut
[5] https://www.google.com/earth/about/versions/

The objects tagged are representative, not exhaustive. Attributes of the objects (e.g., color, size, material) are recorded in the comments field as key:value pairs. The object tags are aligned with their corresponding winery location in the dataset.

3.3.2 Open Street Maps Tags. *GESTALT* can also ingest object and location tags by querying the Open Street Maps (OSM) API [4] 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 this time excluding results that lack features like names, street addresses, and phone numbers. We assign each location or object tag ingested from OSM a confidence score of 0.75 because while any user can edit tags in the OSM database without review, the open-source community self-regulates [19], and prior work has examined the application of machine learning for detecting anomalous behavior in OSM edits [11].

3.3.3 Noisy Image-based Tags. The third and most important method by which *GESTALT* ingests objects is through 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 [6] to identify objects in each image from 80 classes (based on the COCO dataset [7]). Those objects become tags in *GESTALT*, with their coordinates determined by the geolocation of the image and their confidence score determined by YOLO's confidence at the detection step. For our experiments we enable safe-search and limit to public photos. For the Swan Valley Region we retrieve 462 images, and for the Washington DC Region, we pull 4,000 images.

Region	Type	Source	Label Method	# Tags
Swan Valley[8]	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.[9]	Objects	OSM B-Box	Crowd-Sourced	60123
	Objects	Flickr B-Box	Object detection	31065
	Locations	OSM B-Box	Crowd-Sourced	12179

Table 1: Summary of location and object datasets in *GESTALT*. Rows corresponding to the Swan Valley region and having Hand Labels are part of the gold standard Swan Valley Wineries dataset we contribute.

[6]YOLO
[7]https://cocodataset.org/
[8]BoundingBox:['115.96168231510637', '-31.90009882641578', '116.05029961853784', '-31.77307863942101']
[9]BoundingBox:['-77.120248', '38.791086', '-76.911012', '38.995732']

4 OWNERSHIP ASSIGNMENT

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 call this problem *Object Ownership Assignment* and define it as follows. Given a collection of locations and objects within a region, *Ownership Assignment* seeks to assign each object to its 'parent' location correctly. Ownership Assignment amounts to a clustering problem where points (objects) are assigned to centroids (locations) that are known apriori. The objects are not uniformly distributed across locations, and some locations may not have associated tagged objects.

Further, the human eye does 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 out back 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 of the method.

4.1 Object Ownership Assignment Dataset

We evaluate the Ownership Assignment task on two datasets: the hand-labeled Swan Valley Wineries dataset alone 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 from the Flickr dataset and the crowd-sourced object tags from the OSM dataset. 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 on the Combined dataset. Instead, we measure only the recall on the same Winery locations and their known objects that were hand-labeled as part of the Swan Valley dataset. The Combined dataset tests how well the recall performance of our Ownership Assignment method holds up to additional noise and more densely packed objects and locations. Table 2 contains the recall (and precision, where applicable) on those locations and objects.

4.2 Object Ownership Assignment Method

Our method for Object Ownership Assignment is outlined in Algorithm 1. 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 cluster centroids to their nearest known location tags. 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 to be noise and are not relevant or helpful in 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 the centroid (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
```

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. The epsilon value is set to the equivalent of roughly 50m, a reasonable cluster neighborhood size for such an environment. 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.

4.3 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, while straightforward, 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 which do not belong to any locations are relegated to the NULL cluster (i.e., a mistagged singular object with nothing else around it for miles in any direction), and 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.

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.

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 below 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. For the Swan Valley set we also provide the expected results set for each query. The Swan Valley queries are designed to measure precision and recall. To construct them, we hand labeled the ground truth responses for 24 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 10 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).

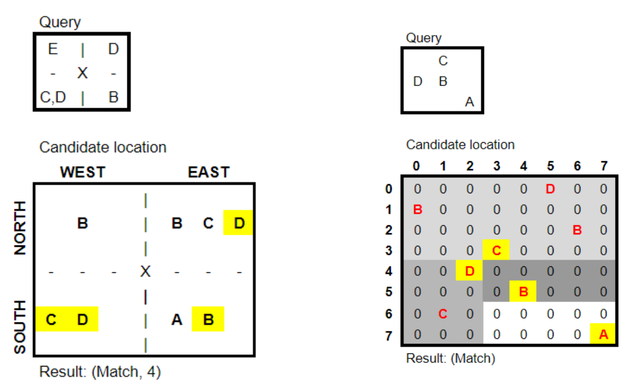
5.2.2 ***Ranked membership search.*** When the exact membership search returns a large number of hits, such as for a broad query (i.e. Which locations have a tree and a bench?) the ranking of those locations can help narrow the results. Using Algorithm 3, we aggregate the confidence scores from the object 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 matching locations, we use a fuzzy search procedure (Algorithm 4) to find a set of partial match locations based on the most discriminative object(s) in the list of query terms provided. The most discriminative terms are emphasized since rare objects are memorable and more uniquely identify locations than common objects do.

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)
  $SearchResult \leftarrow$ **II.SEARCH**(Q)
6: **if** $SearchResult$ is Empty **then**
    $P \leftarrow$ Pop most discriminative term from Q
8:    $SearchResult \leftarrow II$.**SEARCH**($Q$)
    if $SearchResult$ is not Empty **then**
10:      Skip to Line 6
    else
12:      $SearchResult \leftarrow$ **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 4 cardinal directions, depicted in Figure 3a. The object-centric search method (Object-Object search) allows the user to specify the relative layout of objects concerning each other, regardless of where they are relative to the location under search (Figure 3b. A third form of spatial 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 methods using pictorial queries. In Figure 3a, objects are queried by cardinal direction to location sought. In Figure 3b, locations containing objects in a given spatial configuration to each other are sought.

5.3 Search Results

Query Method	Metric	Results
Loc-Obj	Mean Precision	0.854
	Mean Recall	0.917
Obj-Obj	Mean Precision	0.721
	Mean Recall	0.875

Table 3: Spatial results across 24 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 queries* and the *Combined* dataset. The results (Table 3) show that *GESTALT* has a high recall on both methods of query specification, object-location and object-object. We expect precision to be low on these query results when compared to the ground-truth, since the *Combined* dataset includes many additional locations that were not considered during the hand-labeling. However, the precision we recorded is reasonable, which points to the discriminative power of spatial configurations of objects as a search constraint.

Search Complexity. A detailed theoretical analysis of search complexity for the three forms of spatial search is presented in *COMPASS* [13]. To assess 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 takes place 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 is done when the right object is found to the wrong cardinal direction of the location.

6 RELATED WORK

GESTALT takes a human-centric approach to geospatial search, leveraging human spatial cognition, navigation and interpretation of objects on a map. We address related work in human spatial reasoning and pictorial querying, and then compare *GESTALT* with the closest geospatial search tool available currently.

6.1 Human Spatial Cognition and Reasoning

People tend to anchor memories on objects they see while experiencing a location [6], 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 within a region given information about their objects. Last-mile search forms the reciprocal problem to the online game *Geoguessr* [10], which challenges users to determine the broad region of a photograph based on the visible objects. The success of high-performing players of *GeoGuessr* shows 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 [**?**]. *GESTALT*'s approach to geospatial search closely follows the *Winthrop Method* [7] 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 [9]. 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].

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

6.2 Spatial and Pictorial Querying

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. The most common forumulations of the problem are as qualitative spatial reasoning constraint satisfaction problems, set-based search and as a graph-matching problem. Dylla et. al. provided a comprehensive survey of these qualitative reasoning approaches in 2017 [1]. Despite all of the approaches abstracting the real-world for reasoning, none of the systems that they surveyed are performant at a production system scale. Set-based approaches to pictorial querying explored matching on the dimensions of *matching similarity*, *contextual similarity*, and *spatial similariy* [16] [17]. They were then reshaped as tree-based queries [18] and then again as isomorphic subgraph matching problems to strengthen topological matching, at the cost of becoming NP-Complete [3]. Work in *query-by-sketch* is related but distinct area of that typically seeks to match a map area in database by a user sketch-map that is digitised and used as a query. Systems like *Sketchmapia* [14] are better suited for retriving areas larger than individual locations.

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 determine to match or not match the configuration specified in the pictorial query.

6.3 Automated Geospatial Search

In recent months the Open Source Intelligence platform *Bellingcat*[11] released *Bellingcat OSM Search*[12][21], 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 their geographic configuration. *GESTALT* allows users to leverage 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 object tags which give *GESTALT* much of its pruning power.
(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 not very high. *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, taking the position a partial answer is better than no answer.

[11]Bellingcat Website
[12]Bellingcat OSM Search Tool

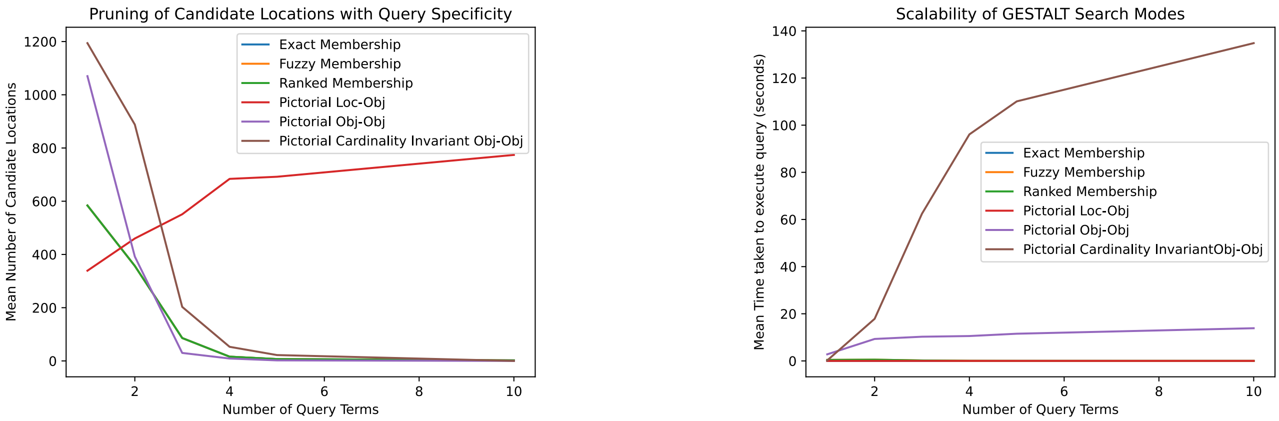


Figure 4: Number of candidate locations and query response times for each type of spatial query *GESTALT* supports. Measured on the Washington D.C. Dataset (12,179 Locations, 91,188 objects).

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 [5, 8, 10, 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.

REFERENCES

[1] Frank Dylla, Jae Hee Lee, Till Mossakowski, Thomas Schneider, André Van Delden, Jasper Van De Ven, and Diedrich Wolter. 2017. A survey of qualitative spatial and temporal calculi: algebraic and computational properties. *ACM Computing Surveys (CSUR)* 50, 1 (2017), 1–39.

[2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei 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.

[3] Andre Folkers, Hanan Samet, and Aya 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.

[4] Mordechai Haklay and Patrick Weber. 2008. Openstreetmap: User-generated street maps. *IEEE Pervasive computing* 7, 4 (2008), 12–18.

[5] James Hays and Alexei A Efros. 2008. IM2GPS: estimating geographic information from a single image. In *2008 ieee conference on computer vision and pattern recognition*. IEEE, 1–8.

[6] Jason Helbing, Dejan Draschkow, and Melissa 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

[7] David Keatley, Chris O'Donnell, Brendan Chapman, and David 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.

[8] Yang Liu, Jie Jiang, Jiahao Sun, Liang Bai, and Qi 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.

[9] Jonathan F Miller, Markus Neufang, Alec Solway, Armin Brandt, Michael Trippel, Irina Mader, Stefan Hefft, Max Merkow, Sean M Polyn, Joshua Jacobs, et al. 2013. Neural activity in human hippocampal formation reveals the spatial context of retrieved memories. *Science* 342, 6162 (2013), 1111–1114.

[10] Yue Ming, Xuyang Meng, Chunxiao Fan, and Hui Yu. 2021. Deep learning for monocular depth estimation: A review. *Neurocomputing* 438 (2021), 14–33.

[11] Peter Mooney, Marco Minghini, et al. 2017. A review of OpenStreetMap data. *Mapping and the citizen sensor* (2017), 37–59.

[12] Miguel Oliveira, Luís Seabra Lopes, Gi Hyun Lim, S. Hamidreza Kasaei, Ana Maria Tomé, and Aneesh 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] Kent O'Sullivan, Nicole R. Schneider, and Hanan Samet. 2023. COMPASS: Cardinal Orientation Manipulation and Pattern-Aware Spatial Search. *Under Review at GeoSearch 2023* (2023).

[14] Angela Schwering, Jia Wang, Malumbo Chipofya, Sahib Jan, Rui Li, and Klaus Broelemann. 2014. SketchMapia: Qualitative representations for the alignment of sketch and metric maps. *Spatial cognition & computation* 14, 3 (2014), 220–254.

[15] Noah Snavely. 2011. Scene reconstruction and visualization from internet photo collections: A survey. *IPSJ Transactions on Computer Vision and Applications* 3 (2011), 44–66.

[16] Aya Soffer and Hanan Samet. 1997. Pictorial query specification for browsing through image databasess. In *Proceedings of the Second International Conference on Visual Information Systems*. 117–124.

[17] Aya Soffer and Hanan Samet. 1998. Pictorial query specification for browsing through spatially referenced image databases. *Journal of Visual Languages & Computing* 9, 6 (1998), 567–596.

[18] Aya Soffer and Hanan Samet. 1999. Query processing and optimization for pictorial query trees. In *International Conference on Advances in Visual Information Systems*. Springer, 60–68.

[19] John E Vargas-Munoz, Shivangi Srivastava, Devis Tuia, and Alexandre X Falcao. 2020. OpenStreetMap: Challenges and opportunities in machine learning and remote sensing. *IEEE Geoscience and Remote Sensing Magazine* 9, 1 (2020), 184–199.

[20] Steven M Weisberg and Nora 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.

[21] Logan Williams. 2023. Finding Geolocation Leads with Bellingcat's OpenStreetMap Search Tool. *Bellingcat Blog* (May 2023).