Street-Level Geolocation From Natural Language Descriptions

Nate Blaylock* — **James Allen**** — **William de Beaumont**** — **Lucian Galescu**** — **Hyuckchul Jung*****

* *Nuance Communications*
nate.blaylock@nuance.com

** *Florida Institute for Human and Machine Cognition (IHMC)*
{jallen,wbeaumont,lgalescu,hjung}@ihmc.us

*** *AT&T Labs - Research, Shannon Laboratory*
hjung@research.att.com

ABSTRACT. *In this article, we describe the TEGUS system for mining geospatial path data from natural language descriptions. TEGUS uses natural language processing and geospatial databases to recover path coordinates from user descriptions of paths at street level. We also describe the PURSUIT Corpus — an annotated corpus of geospatial path descriptions in spoken natural language. PURSUIT includes the spoken path descriptions along with a synchronized GPS track of the path actually taken. Finally, we describe the performance of several variations of TEGUS (based on graph reasoning, particle filtering, and dialog) on PURSUIT.*

RÉSUMÉ. *Dans cet article, nous décrivons le système TEGUS destiné à la fouille de données sur des trajectoires géospatiales, à partir de descriptions en langage naturel. TEGUS utilise des techniques de Traitement Automatique des Langues et des bases de données géospatiales pour retrouver les coordonnées de trajectoires à partir des descriptions faites par des utilisateurs de leur chemins au niveau des rues. Nous décrivons également le corpus PURSUIT, un corpus annoté de descriptions de chemins géospatiaux en langage parlé. PURSUIT inclut les descriptions de chemins avec la trace GPS du chemin effectivement suivi. Enfin, nous décrivons les performances de plusieurs variations de TEGUS (avec raisonnement sur des graphes, filtre de particules, gestion de dialogue) sur le corpus PURSUIT.*

KEYWORDS: *geospatial information extraction; geospatial path understanding.*

MOTS-CLÉS : *extraction d'information géospatiale; compréhension de trajectoire géospatiale.*

1. Introduction

We are interested in building algorithms that understand natural language (NL) descriptions of spatial locations, orientation, movement and paths that are grounded in the real world. Our ultimate goal is to be able to extract such information from arbitrary text or speech. In this paper, we describe our work on the problem of *geospatial path understanding*: extracting a path in latitude and longitude (lat/lon) coordinates given a natural language description of that path. Path understanding would enable a number of applications, including automated geotagging of text and speech, robots that can follow human route instructions, and geolocation without the use of GPS.

Our work is centered around geospatial path understanding at *street level* — the level of such entities as streets, intersections, businesses, parks, etc. This is in contrast to most work in geospatial language understanding, which has been at the level of larger entities such as cities, states, and countries.

Our approach to this problem is based on using natural language understanding to extract references to geospatial entities. These linguistic references are used to formulate queries to large geospatial databases to search for possible grounding *referents*. Both reference extraction and search are noisy processes, which produce uncertain results. However, we are able to use contextual information (such as timing between utterances) to improve performance.

The remainder of this paper is structured as follows. First, we detail related work in this area. We then describe the geospatial databases we use in this work. We then describe the corpus we use for training and testing. We then detail three different versions of a path geolocation system and its experimental results. We conclude by mentioning future directions.

2. Related Work

Although there has been a fair amount of previous work on geospatial reference extraction and resolution (see, *inter alia*, Leidner *et al.*, 2003; Mani *et al.*, 2008; Mikheev *et al.*, 1999) most has been at the level of cities, states, and countries. Our work focuses on street-level references to entities such as streets, intersections, and businesses. The street level contains a much larger set of possible referents and contains much more name ambiguity than the city/state/country level. Additionally, there seems to be a lot more variety in the way geospatial entities are referred to at the street level, including reference by category. Finally, geospatial databases are much less complete than they are for information about cities, states and countries, due both to the closed-set nature of the latter as well as differences in the pace of change (some businesses referred to in our corpus below have since moved, gone out of business, or changed names).

We first mention related work in the area of geospatial corpora. We then turn our attention to related work in geospatial language understanding.

2.1. *Geospatial Corpora*

Although corpora exist for studying NL path descriptions, we are not aware of any that are bundled with the corresponding GPS track for the paths. In addition, many corpora are in the *spatial* domain, but not the *geospatial* domain (i.e., they do not contain references to real-world entities with lat/lon coordinates). For example, in the Map Task Corpus (Anderson *et al.*, 1991), paths described were drawn on 2-D maps of a fictitious world with relatively few landmarks and no streets. Even setting aside the fact that it is a fictitious world, the number of geospatial entities and complexity of the task is much different in a real-world urban environment.

The MARCO corpus (MacMahon *et al.*, 2006) describes paths through a 3-D virtual world of indoor corridors. The problem of indoor navigation (such as in malls, airports or other large buildings) is of interest to us, and the techniques described in this paper could be applicable to it, but the level of publicly available data is currently very low. The IBL corpus (Bugmann *et al.*, 2004) contains path descriptions of robot movement through a miniature (fictitious) town model. Neither of these are directly applicable to geospatial databases since each is in a fictitious environment and, with the exception of Map Task, movement on each is on a small scale. The smallest objects in our geospatial database (as we will outline below) are at the scale of buildings — thus the scale of the path needs to be on the order of hundreds of meters so that multiple GIS objects might be referenced.

SpatialML (Mani *et al.*, 2008) is an XML-based markup language for annotating geospatial references in text. It can represent references to geospatial entities as well as certain spatial relations between them. However, as far as we are aware, there are no SpatialML corpora that have been created at the street level.

ISO-Space (Pustejovsky *et al.*, 2011) is a successor to SpatialML that also allows the annotation of orientation, movement, and geospatial paths. This seems very compatible with the goals of our own work. ISO-Space is still under development, however, and we are not aware of any corpora annotated with it.

2.2. *Geospatial Language Understanding*

As far as we are aware, very little work has been done in the area of mining geospatial path data from descriptions. The work in Pustejovsky and Moszkowicz (2008) uses lexical verb mappings to a spatial calculus towards extracting paths from text descriptions. The work in Zhang *et al.* (2010) explores webpages containing human-created route descriptions and extracts the final destination using a number of HTML and linguistic features, towards being able to automatically interpret the entire route instructions.

The MUTTS system (Schmill and Oates, 2011) is very similar to our own system described below. It parses natural language descriptions of paths and uses a particle filter to geolocate that path. MUTTS extracts templates of spatial relations and

movement from the parse, along with temporal references, and uses them to update a particle filter that models possible locations.

The work in Sledge and Keller (2009) uses histograms of force to match a text route description to building outlines extracted from satellite imagery. It parses a text description and then maps it onto a very precise set of spatial operators (such as "perfectly to the left of") which is used to specify the histograms of force. These then settle on actual building outlines in the area — linking both the path and the mentioned landmarks.

3. Geospatial Databases

Before describing our geospatial work, we will go over the data sources and databases used in the research detailed in the rest of the paper.

Geospatial databases are critical in our approach, as they are used to tie names to geospatial entities which have a latitude and longitude component. In our work, we use two geospatial databases: Google Maps and TerraFly.

3.1. *Google Maps*

Google Maps [1] is the online mapping service provided by Google, which we access through a RESTful Web interface. The Google Maps API supports searching for matches on a string query near a central lat/lon coordinate (and within a bounding box). Determining matches to the search string is based on a proprietary, unpublished Google algorithm which appears to match tokens both in the names of geospatial entities as well as tokens in their descriptions.

The contents and provenance of the Google Maps geospatial database are not publicly known. From our experience, the RESTful API only returns point data such as businesses and other points of interest. It does not allow access to street or intersection data.

3.2. *TerraFly*

TerraFly (Rishe *et al.*, 2005) is a geospatial database developed at Florida International University, and comprises a number of geospatial data sources. We accessed TerraFly via the Spatial Keyword Search (SKS) (Felipe *et al.*, 2008) Web interface. SKS supports boolean, token-based string search on any given field in the database for matches within a radius of a given lat/lon coordinate.

1. http://maps.google.com.

In our work, we used three separate TerraFly databases:

Streets and street segments Typically, geospatial data about streets is stored as raster data in geospatial information systems (GIS). For our purposes, however, we use a TerraFly database in which street segments are represented as point data. (A street segment is typically a portion of a street between intersections or other waypoints.) This allows us to reason with streets as if they were single points. In the future work section, we describe needs for more sophisticated geospatial reasoners that would include support for lines and polygons.

TerraFly also includes a street database, which includes a set of street segments for a given street name. As ontological objects, streets are a bit complicated. A given street segment may be part of several different street names, as named streets and route indicators (such as US-90) often overlap. This database is used to provide grounding references to mentions of streets in text.

The street databases we used were derived from the commercial NAVSTREETS route dataset from NAVTEQ. [2]

Intersections We also used a database of street intersections, compiled from the street database mentioned above. Each intersection is represented by a name, and lat/lon, and a list of streets that intersect at it. There are several things to mention here: first, as some pairs of streets intersect more than once, a lat/lon coordinate is needed to uniquely identify an intersection. Also, the list of intersecting streets may be greater than two because (a) some intersections are comprised of more than 2 physical intersecting streets; (b) the same contiguous street may have different names on either side of an intersection; and (c) the intersecting street segments may have more than one name (as mentioned above).

Other point data The final TerraFly database we used was a compendium of data for entities that are not streets or intersections, such as businesses, bridges, parks, schools, restaurants, hotels, bodies of water, etc. These were aggregated from several datasets including: the Geographic Names Information System (GNIS), NAVTEQ Points of Interest (POI), Infot Business Databases, Yellow Pages, and US Census data. The data was ameliorated and then heuristically deduplicated. Access to a single point database provided a canonical data source for grounding references in data (which is described in more detail in the next section).

Additionally, we used a land parcel-based geocoder from TerraFly to convert street addresses to lat/lon coordinates.

4. PURSUIT Corpus

To aid the development and evaluation of our path geolocation systems (described below), we developed the PURSUIT Corpus (Blaylock and Allen, 2008; Blaylock, 2011), which consists of 13 audio recordings of spoken path descriptions that were made in realtime as the path was driven in an urban area. Additionally, the corpus

2. http://www.navteq.com.

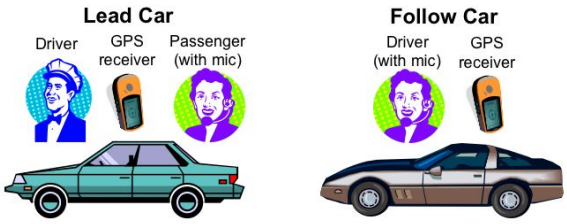

Figure 1. *Data collection setup*

includes corresponding synchronized GPS tracks for each recording, which represent the "ground truth" of the path actually traversed.

The corpus has been manually transcribed, segmented, and annotated with geospatial entity references. The synchronized combination of information from the speech and GPS "modalities" has allowed us in a fairly reliable way to manually identify the intended real-world geospatial entities referred to by mentions in the speech.

In the following, we describe the data collection method for the corpus and the various annotations performed on it. In particular, we detail our strategy for semantic annotation of entities based on a combination of name, address, and lat/lon coordinates. The PURSUIT Corpus is freely available for download at http://www.cs.rochester.edu/research/speech/pursuit/.

4.1. *Corpus Data Collection*

Figure 1 shows an example of the data collection setup for the corpus collection. Each session consisted of a lead car and a follow car in downtown Pensacola, Florida. The driver of the lead car was instructed to drive wherever he wanted for an approximate amount of time (around 15 minutes). The driver of the follow car was instructed to follow the lead car. One person in the lead car (usually a passenger) and one person in the follow car (usually the driver) were given close-speaking headset microphones and instructed to describe, during the ride, where the lead car was going, as if they were speaking to someone in a remote location who was trying to follow the car on a map. The speakers were also instructed to try to be verbose, and that they did not need to restrict themselves to street names — they could use businesses, landmarks, or whatever was natural. Both speakers' speech was recorded during the session. In addition, a GPS receiver was placed in each car and the GPS track was recorded at a high sampling rate.

	Corpus	Avg. per Session Track
Length	3h55m	18m
Utterances	3,155	243
Words	20,228	1,556
Annotated References	1,649	127

Table 1. *PURSUIT Corpus statistics*

4.1.1. *Data*

The corpus contains 13 audio recordings of seven paths along with two corresponding GPS tracks for each path, from the two cars. [3] The average session length was just over 18 minutes, and overall 1,649 geospatial references were annotated. Table 1 shows various information about the corpus size.

The corpus is rich with references to geospatial entities. Some sample utterances from the corpus are given below:

– ... and we're going under the I-110 overpass I believe and the Civic Center is on the right side on the corner of Alcaniz and East Gregory Street where we are going to be taking a left turn...

– ... he's going to turn left right here by the UWF Small Business Development Center heading toward Gulf Power...

– ... we've stopped at a red light at Tarragona Street okay we're going now across Tarragona passing the Music House...

– ... we're at the intersection of East Gregory and 9th near a restaurant called Carrabba's I think and a Shell station just a little south of the railway crossing...

4.2. *Annotation*

The corpus has been manually annotated with transcription, utterance, and location reference information. Below we describe the annotation format, tools we developed for the annotation and visualization of the data, and specific issues that emerged during the development of the corpus.

4.2.1. *Annotation Format*

We use the NITE XML Toolkit (NXT) data model (Carletta *et al.*, 2005) for storing both the corpus and annotations on it. NXT is a general XML data model for multimodal and heavily cross-annotated corpora. In the data model, a corpus is represented as a list of observations, which contain the data for a single session. An observation

3. In one session only one audio recording was made.

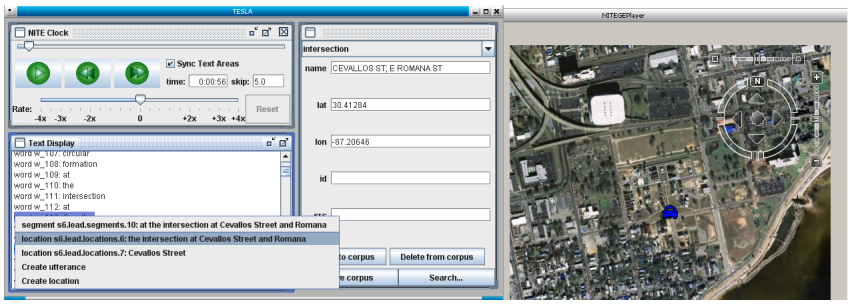

Figure 2. *Main view of the TESLA annotator*

contains a set of synchronized signals, which are typically audio or video streams associated with the observation, although NXT is broad enough that a signal may be any timestamped stream of data (like our GPS tracks). Annotations are represented as a multi-rooted tree structure, where leaves are segments that are time-aligned with an underlying signal. This allows disparate annotations to be made on and saved with the same corpus.

4.2.2. *The TESLA Annotation and Visualization Tool*

To annotate the PURSUIT Corpus, we built The gEoSpatial Language Annotator (TESLA) — a tool that supports the visualization and hand-annotation of both text and speech-based geospatial language corpora (Blaylock *et al.*, 2009). TESLA can be used to create a gold-standard for training and testing geospatial language understanding algorithms by allowing the user to annotate references to geospatial entities and lat/lon coordinates. An integrated search capability to geospatial databases with results presented in Google Earth allow the human annotator to easily annotate geospatial references with ground truth. Furthermore, TESLA supports the playback of GPS tracks of multiple objects for corpora associated with synchronized speaker or object movement, allowing the annotator to take this positional context into account.

Figure 2 shows a screenshot of the main view in the TESLA annotator, showing a session of the PURSUIT Corpus. In the top-left corner is a widget with playback controls for the session. This provides synchronized playback of the speech and GPS tracks. When the session is playing, audio from a single speaker (lead or follow) is played back, and the blue car icon in the Google Earth window on the right moves in synchronized fashion. Although this Google Earth playback is somewhat analogous to a video of the movement, Google Earth remains usable and the user can move the display or zoom in and out as desired. If location annotations have previously been made, these pop up at the given lat/lon as they are mentioned in the audio, which allows for easy verification (and correction, if necessary) by the annotator. In the center, on the left-hand side is a display of the audio transcription, which also moves in sync with the audio and Google Earth visualization. The user creates an annotation by highlight-

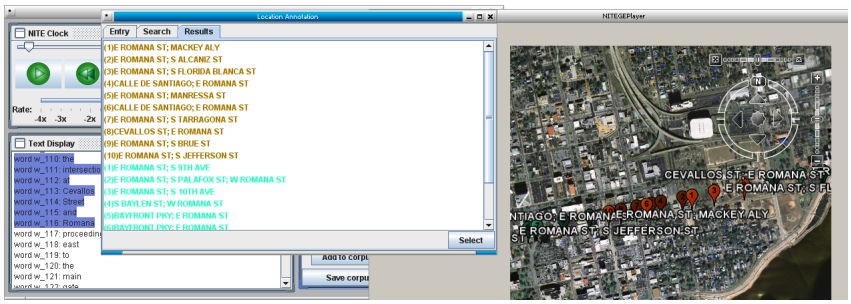

Figure 3. *Search results display in TESLA*

ing a group of words, and choosing the appropriate type of annotation. The currently selected annotation appears to the right where the corresponding geospatial entity information (e.g., name, address, lat/lon) can be entered by hand, or by searching for the entity in a geospatial database.

In addition to allowing information on annotated geospatial entities to be entered by hand, TESLA also supports search with geospatial databases. TESLA, by default, uses the position of the GPS track of the car at the time of the utterance as the center for search queries, although any point can be chosen.

Search results are shown to the user in Google Earth as illustrated in Figure 3. This figure shows the result of searching for intersections with the keyword "Romana". The annotator can then select one of the search results, which will automatically populate the geospatial entity information for that annotation. Such visualization is important in geospatial language annotation, as it helps the annotator to verify that the *correct* entity is chosen.

Since there will be ambiguous references in the corpus (such as "the gas station"), it is important that we include this feature to help the user visually disambiguate the reference. In addition, it is possible that the result will seem unambiguous (e.g., only one result is returned), but the seemingly unambiguous result is incorrect (e.g., the result is all the way across town from the speaker); by displaying the search result(s) in Google Earth, we avoid accidentally falling into such a situation.

4.2.3. *Synchronization*

For each session, the resulting two audio and two GPS track files were synchronized by hand to start and end at the same point in time. As the recording on each device was started separately from the others, this led to special challenges in synchronization. Using TESLA, the human annotator adjusted audio and GPS length and starting time by hand until the audio descriptions and GPS tracks were in concordance.

4.2.4. *Transcription*

Transcription of the audio signal was done manually using the Transcriber tool (Barras *et al.*, 2000). The resulting transcription included not only words, but also preliminary utterance breaks that were useful to the transcriber. These were used later to estimate word timing information, as discussed below.

Transcription rules were that no punctuation was to be transcribed, except in phrases requiring a hyphen, periods in names with abbreviations, and apostrophes. Proper nouns were capitalized, but the beginnings of utterances were not. Internet resources such as Google Maps were used to verify canonical spellings of proper nouns such as business or street names. Numbered street names were spelled out (e.g., "Seventeenth Avenue"). In cases where the correct transcription could not be determined, the token [`unintelligible`] was inserted as a word.

The words level in NXT requires not only the list of transcribed words, but also timing information on the start and end time of each word. This was estimated by using the rough Transcriber utterance boundaries for the start and end time of each rough utterance and equally dividing the utterance time into chunks for each word within it.

As we will discuss below, the timing information in the corpus is quite important in this domain, as it places a constraint on possible distances moved in the given time. For example, if a speaker mentions a park in one utterance and then 10 seconds later mentions an intersection, we can assume that the car cannot have moved 5 miles during that time.

4.2.5. *Utterance Segmentation*

Utterance segmentation was done manually using TESLA. Utterance segments of spoken monologue are admittedly somewhat arbitrary, but annotators were instructed to use cues such as pauses and grammar to help determine natural utterance breaks.

4.2.6. *Geospatial Reference Annotation*

References to certain types of locations were segmented and annotated by hand with information about each referent using the TESLA tool.

The high-level classes annotated were:

– *Streets*: references to a given street, for example "Garden Street" or "a divided road";

– *Intersections*: references to street intersections, for example "the corner of 9th and Cervantes" or "the next intersection";

– *Addresses*: references to street address, for example "401 East Chase Street" or even "712" (when referring to the address by just the street number);

– *Other Locations*: this class is a grab bag for all other location types that we annotated, consisting of such data as businesses, parks, bridges, bodies of water, etc.

This classification was chosen because that was the separation of data types in our geospatial databases (as noted above), and not for deep ontological reasons. We do believe, however, that the standardization of a geospatial entity ontology is needed.

Note that not all geospatial entity references have been annotated in PURSUIT — just those *types* that are accessible in our geospatial databases. Examples of entities referred to in the corpus that were not annotated are fields, parking lots, sidewalks, railroad tracks, neighborhoods, and fire hydrants. These were not annotated only because we did not have access to data about those entities. However, there is nothing inherent in our approach to path understanding which would prohibit the use of those classes of entities, if data were available for them. Indeed, we believe that much more geospatial data will be made available in the not-too-distant future through the release of private or government databases, advanced mining techniques (e.g., Knoblock *et al.*, 2010), and geospatial crowdsourcing.

Although not all *classes* of entities were annotated, within those classes that were annotated, *all* references to entities of interest were annotated, whether or not they were actually found in the geospatial databases. Additionally, all references to entities were annotated, including category and pronoun references. Thus "Garden Street", "a divided road", or even "it" were annotated if they referred to a geospatial entity of interest. Each entity was also annotated with whether an entity reference was *named* (i.e., contained at least part of the proper name of the entity, such as "the Music House" and "the intersection at Cervantes") or *category* (description did not include a name, such as "the street", "a Mexican restaurant", and "it").

Annotators were instructed to bracket the entire referring phrase, not just the headword, as is done in SpatialML (Mani *et al.*, 2008). One reason for this is that it allowed the annotation to reflect embedded references. For example, many references to intersections also mention streets. The phrase "the corner of 9th and Cervantes" contains references to an intersection and two streets. Although it would be possible to just annotate the headwords (e.g., *corner*, *9th*, and *Cervantes*), that annotation loses the information that, indeed, the intersection is at these two roads.

In total, 1,649 geospatial entity references were annotated in the corpus. The breakdown by category is shown in Table 2.

4.2.7. *Grounding Geospatial References*

Although the manual bracketing of references is relatively easy, deciding which real-world entities they correspond to is not, in general. Additionally, as the set of geospatial entities at the street-level is not closed, there is a question as to how to represent the real-world entities as semantic individuals.

4.2.7.1. Semantic Representation

In an ideal world, we would have access to a single knowledge base which contained all possible geospatial entities with unique IDs that we could use to ground geospatial references. Our reality is that we had two geospatial point databases with

	Reference Type		
	Named	**Category**	**Total**
Street	77.2%	22.8%	**48.5%**
Intersection	45.5%	54.5%	**6.8%**
Address	100.0%	0.0%	**0.8%**
Other Loc	67.7%	32.3%	**43.9%**
Total	**71.1%**	**28.9%**	**100%**

Table 2. *Breakdown of geospatial entity reference annotations in the PURSUIT Corpus*

varying coverage, and, in the case of Google, with no direct access to the underlying dataset. In fact, out of 724 *other_loc* references, 25.7% were in both databases, 16.7% were only in TerraFly, 40.1% were only in the Google Maps database, and 17.5% were in neither.

Latitude/longitude coordinates alone are also not a viable way to uniquely identify a geospatial entity. First, lat/lon coordinates, represented as decimals, have arbitrary precision. One dataset may represent lat/lons to the thousandths place, whereas another to the hundred-thousandths, making it impossible to know if two lat/lon coordinates refer to the same entity (remember, our goal is to ground *entities*, not locations). Second, although represented as point data, most geospatial entity data is actually 2-dimensional — a business may refer to the lot it is on, instead of an arbitrary point on that property. Third, many databases are geolocated automatically based on street address (where available). In our experience, many times the given lat/lon can be up to 200 meters from the actual location. It can be worse in rural areas. Different datasets may have different geolocations, and there is no trivial way to determine if two lat/lon coordinates refer to the same entity. Lastly, consider the case where several businesses reside at a single address, such as in a shopping mall. A dataset may have several entities for a single lat/lon.

Using the entity name as a unique ID is also problematic, as an entity may have several aliases or may be referred to in different ways — for example, *IBM*, *I.B.M.*, *IBM Corp.*, *International Business Machines*, etc.

Although we do not have a perfect solution, we outline the approximation we have taken for the different types of entities.

Other Locs Our catch-all *other_loc* class contains entities such as businesses, parks, bodies of water, etc. Each is annotated minimally with a canonical name (from the database, if available, or chosen by the annotator based on internet searches) and a lat/lon (from the database, or manually chosen by the annotator on Google Earth). For entities which have an address, this is added as well.

Streets We annotated each street reference to the nearest street segment terminus to the user's location, with the given name of the street referred to.

Intersections Intersections are represented as in the TerraFly database, as described above.

Addresses Addresses are represented by their full street address.

4.2.7.2. Grounding

Entities were annotated using the TESLA tool, including the geospatial search capability described above. In cases where the entity was not found in a databases, annotators had good local knowledge of the area, and with this, in many cases were able to identify the intended referent. If that was not enough, annotators would sometimes use Google Street view to look at images at the location. In a small number of cases, however, an annotator had to physically travel to the given location in order to understand and get information about entities referred to in the corpus.

4.2.7.3. Source Database Information

As noted above, several sources were used to search for geospatial entity information for annotation. The data sources are also noted in the annotation on each reference. The two main data sources used are TerraFly and Google Maps (which we will describe in more detail below). Entities which were not available in either data source are marked correspondingly. Overall, 92.2% of geospatial entity references were in either or both of the geospatial databases used.

5. *Post Hoc* Path Geolocation

We now turn to the task of path geolocation and how we approached it in our path understanding system: TEGUS (ThE Geospatial language Understanding System). In this section, we describe a version of TEGUS that performs *post hoc* path geolocation, i.e., geolocation given the entire description of the path. In the next section, we relax that constraint and describe a version of TEGUS which performs *online* path geolocation — making a location prediction after each user utterance. In the subsequent section, we describe a version of TEGUS which transforms this into a dialog system in which the user and system collaborate to more quickly determine the user's current location.

5.1. *System Description*

Input to the system is the natural language text description of a path. The description is parsed to a Logical Form (LF), from which references to geospatial entities are extracted. These references are used to generate queries to the two geospatial databases, producing a list of possible entity matches. The list of references and pos-

sible matches are then used in a path finding algorithm which uses temporal (i.e., utterance timing) and geospatial constraints to predict the most likely path.

Note that, in the current versions of the system, the path that is recognized is not based on traversal of a street network. Rather, the assumption is made that geospatial entities being mentioned will be in the close vicinity of the current position of the user. The "path" predicted by the system is actually an ordered list of lat/lon coordinates.

It is also important to note that, although GPS track information was gathered as part of the corpus, it is *not* used as input to the path understanding system. The only input to TEGUS is the transcribed text description of the path in natural language. The GPS track information is used as ground truth for evaluation, as explained below.

We now describe each of the system's subcomponents.

5.1.1. *Language Processing*

Language processing is done by the TRIPS Parser system (Allen *et al.*, 2008), which is the language understanding component of the TRIPS dialog system (Jung *et al.*, 2008). TRIPS performs deep syntactic/semantic analysis on the text using a hybrid symbolic-statistical model. The result of analysis is an underspecified Logical Form (LF) (Manshadi *et al.*, 2008) which is imported into OWL as an individual (Blaylock *et al.*, 2011). Several external components are used to guide the symbolic parser, which helps ensure broad coverage. These include a statistical shallow parser, several large lexical resources including WordNet, and the Stanford Named Entity Recognizer (NER) (Finkel *et al.*, 2005). During processing, outputs from these components are used as *suggestions* to the TRIPS parser. However, the parser takes a larger context into account and it is free to heed or ignore these suggestions. A detailed description of the TRIPS Parser system is outside the scope of this paper. However, we mention the Stanford NER especially, as it is used to customize the parser in our experiments described below.

5.1.2. *Location Reference Extraction*

LFs for the utterances are then passed to the Entity Extractor. This component uses hand-built, semantic graph matching rules for SQWRL (O'Connor and Das, 2009) to find the subgraphs of the LF that correspond to location references. For each utterance, the set of location reference LF subgraphs are passed to the geospatial search component.

5.1.3. *Geospatial Search*

TEGUS currently accesses only the TerraFly and Google Maps databases, although the architecture allows any number of databases that may be available. As noted above, both databases support queries for keywords and a lat/lon for the center of the search. The system currently makes the simplifying assumption that we know a bounding circle (of 3 miles) where the user is located (which is meant to roughly

correspond to knowing a general location within a city). Below we discuss plans to relax that assumption.

The Search Controller converts each extracted location reference into one or more keyword searches to the geospatial databases above. Except for cases where the LF clearly marks the high-level class of the entity (e.g., such as knowing that the referent is a street), queries are sent to all of the databases. Search results for each query are aggregated and deduplicated based on name and lat/lon coordinates.

5.1.4. *Path Finding*

For each extracted location reference, the Search Controller returns a (possibly empty) list of search results, which represent the potential geospatial referents for the reference. In the *post hoc* version of TEGUS, we process all utterances from the path description to generate a list of location references together with the list of their possible referents. The Path Finder component uses global context to choose the best referent for each location reference. This is done by constructing a directed graph of all possible referents for all location references in the session and finding the optimal path between the start and end points. Although we formulate the Path Finder as a graph search algorithm, it is important to note that geometric distance is not the only factor in determining edge weights in our graph. We are not literally trying to find the shortest path (in the physical sense), but rather the best fit for the given linguistic description.

The search graph is constructed with each possible referent as a node, and edges connecting all nodes from one location reference to the next. This simple graph forms a trellis, and the optimal path through the trellis visits exactly one node for each location reference, which we can then use to predict the referent for that reference. Unfortunately, a trellis alone will not work for this problem. The assumption that each layer contains the correct referent is too rigid, as it is possible that *none* of the entities in the list is the correct answer for the location reference. This can (and does) happen because of problems at any stage of processing. For example, a non-location reference may be incorrectly extracted, or the geospatial databases may not contain the actual referent. Whatever the reason, we need to be able to handle the case where no node is the correct answer for a given location reference.

To do this, we add additional edges from a node to all nodes up to N references ahead of it. This allows the optimal path algorithm to skip a location reference (thus not making a prediction about that node). Although this multiplies the number of edges in the graph, the constant "skip" factor (N) keeps this number manageable. Optimal paths through the graphs produced in our experiments were found quickly. The experimental results reported here were attained by allowing the algorithm to skip up to 10 location references (i.e., $N = 10$).

Next in the graph construction process, TEGUS uses temporal and geometric knowledge to delete any edges that are considered impossible transitions. Because the corpus is based on speech, we are able to preserve the timing information of utter-

ances. Specifically, we know approximately how much time has passed between two location references. Using this information, together with knowledge of limitations on the mode of transportation (a car), TEGUS can determine whether the (physical) distance between two possible referents was possible given a maximum assumed car speed (100 km/h). Remaining edges are weighted by the distance between the two entities plus a penalty for each location reference "skipped" by the edge.

The remaining edges are assigned a weight based on physical distance as well as name similarity. The latter is currently a simple average of similar word tokens between the two entities. As discussed below, we believe more sophisticated similarity measures will help increase accuracy.

Also, for these experiments, we made the simplifying assumption that we knew the start and end locations for the path. Thus, a single node with the correct location was added at the start and the end of the trellis. In experiments with other versions of our system, we remove this simplifying assumption.

Once the graph is constructed, we use Dijkstra's algorithm (Dijkstra, 1959) to compute the least cost path through the graph. This path represents the system's prediction of the actual path taken.

5.2. *Experimental Results*

We performed an evaluation of the *post hoc* TEGUS system on the PURSUIT Corpus. Specifically, we performed a 7-fold cross validation (one for each session) of TEGUS on the PURSUIT Corpus. The Stanford NER was trained on all *named* location references from the remaining 6 sessions and used with the TRIPS parser. Each path description was separately input to the TEGUS system, which produced a single path prediction.

We evaluate the results of this experiment in two ways, using the GPS track and hand-annotations of the corpus. The first evaluation is on the path prediction itself. As noted above, the current version of the system does path prediction by location references as opposed to street segments. Except for street mentions, very rarely will the speaker have been exactly on the lat/lon coordinates of the geospatial entities he mentions. However, our assumption, as explained above, is that these entities will often be very close to the speaker. We check correctness of the predicted path by looking at the distance between the predicted location and the position of the car (as given by the GPS track) at the time the location is referred to in speech. If the distance is within 300 m, it is counted as a correct prediction. [4]

As the algorithm counts on location mentions to predict current location, it is unreasonable to expect that the system will make predictions at the same rate the GPS

4. The allowable distance from the path will ultimately depend on the end application using geospatial path understanding.

	Named		Category		Total	
	Prec.	**Recall**	**Prec.**	**Recall**	**Prec.**	**Recall**
Street	96.8%	73.9%	100.0%	1.1%	**96.8%**	**57.3%**
Intersection	0.0%	0.0%	0.0%	0.0%	**0.0%**	**0.0%**
Address	0.0%	0.0%	0.0%	0.0%	**0.0%**	**0.0%**
Other Loc	86.2%	60.0%	43.5%	11.5%	**79.7%**	**44.3%**
Total	**92.4%**	**64.0%**	**45.3%**	**6.1%**	**88.9%**	**47.2%**

Table 3. *Breakdown of reference resolution results by entity and reference type*

location was sensed (e.g., 1 Hz). Instead, we take as the maximum number of expected predictions the number of location references (from the hand-annotated corpus). This is used in computing recall for the path prediction.

Besides the final path result, we also evaluate the system's performance on correctly resolving each location reference to the corresponding referent as annotated in the corpus (the actual geospatial entity). Here, we also measure precision and recall. Also, similar to what we do with the path, we allow some distance between the predicted and actual points. As we are working with different databases, because of the inaccuracies of geocoding or other errors, entries for the same entity in different databases can have different names and/or lat/lon coordinates. Because of this, we count as correct all system's predictions that are within 200 m of the lat/lon coordinates as annotated in the PURSUIT Corpus. This number was chosen based on anecdotal evidence from examples where the same address was geocoded to different lat/lon coordinates in Google Maps and TerraFly. Also, street references were considered correct based on the named street mentioned, and not the exact street segment as annotated in the corpus, since, for reference resolution, the whole street was referenced by the speaker, not a particular segment.

Results on the test corpus for *path prediction* were **92.6% precision** and **49.2% recall** (64.3% f-score). Results on geospatial entity *reference resolution* were **88.9% precision** and **47.2% recall** (61.7% f-score).

Reference resolution statistics broken down by reference and entity type are shown in Table 3.

5.3. *Discussion*

All in all, the results seem very good, especially the precision in path prediction. Recall numbers are only in the 40's, although it is important to remember what this means in this domain. To get 100% recall would mean to correctly predict the location after the user's first utterance. Given the ambiguities involved in geospatial language,

that kind of performance is probably unrealistic. It is also important to note that, for most extraction tasks, we assume that precision will be far more important than recall.

Many of the missed cases in recall are actually quite difficult, if not impossible, to do based on this graph-based model of location-based path prediction. References such as "the road" or "this intersection" hold very little informational content, making search for them nearly impossible (without returning *all* the street segments or intersections in the search radius). We believe that moving to a street-based modeling will lead to much higher rates of recall.

In reference resolution, resolution for street references was quite high with 96.8% precision, and *other loc* prediction has 88.9% precision. TEGUS does not yet try to extract intersection or address references (which is why no predictions were made), but we include them for completeness.

The performance on named versus category references is quite stark. Overall, for named references (references with at least one proper name in them), TEGUS achieved 92.4% precision and 64.0% recall. For category references (references with no proper names), however, the system only achieved 45.3% precision and 6.1% recall. This is attributable to several factors, including the "the road" type references with little informational content mentioned above, and the fact that category references (such as "a bank") usually match a much wider range of entities than a named reference (such as "Wells Fargo") does.

It is also interesting to note that the precision result for path prediction is almost 5% higher than precision on reference resolution. What seems to be happening here is that, in some cases, the algorithm is predicting an entity which is *not* the intended referent, but nonetheless is still close to the car position. This is partly due to the fact that entities may have similar names in similar areas (e.g., *Alcaniz Street* and *Atelier Alcaniz* [which is a shop on Alcaniz Street]). Category-based descriptions are sometimes also clumped together (e.g., many banks or restaurants in the same vicinity). Additionally, Google Maps often returns results which not only match the keywords from the search, but also entities that are in the general vicinity from those entities. In these cases, the path finding algorithm is faced with a choice among several entities that are very close to each other. We believe that weighting edges based on a better name similarity metric will help here.

As mentioned above, there are several simplifying assumptions that undoubtedly help the results. The assumption that we know the start and end locations of the path is unrealistic in most cases and has been removed in experiments with other versions of the system.

6. Online Path Geolocation

In this section, we describe a version of TEGUS which makes location predictions after each user utterance. This system uses the same components as the *post hoc*

version of TEGUS, except for the graph-based Path Finder. We describe here the system differences and then experiments based on this system.

6.1. *System Description*

For online prediction, we use particle filtering, an efficient method often used in mobile robot localization. Our particle filtering approach models a vehicle's position as a state (i.e., a particle) and its action model that computes next states is based on vehicle's movement with constraints on possible next states confined on roads (and assuming maximum speeds). Sensors are inputs from user's natural language description that contains geospatial references (e.g., "I'm passing Romana street", "I see Bank of America loan offices", etc.). In our system, sensing is passive in that the system gets sensor inputs only when a user provides inputs: therefore, particle computation is performed at irregular intervals, only triggered with user inputs.

6.1.1. *Particle Filter*

Below is a brief description of particle filtering steps used in TEGUS:

Setting: Number of particles = M; Weight sum = 1.0

Initialization: Given the first utterance, make particles at the locations (lat/lon coordinates) of search results for all geospatial references extracted from the utterance. If more particles are needed, create additional particles at random locations within a certain close distance from (randomly chosen) geospatial referents and, in the opposite case, randomly remove particles.

– *When more particles were needed*: A large portion (e.g., 90%) of the weight sum is evenly distributed to particles that exactly correspond to geospatial referents and the remaining weight sum is evenly distributed to additional created particles.

– *Otherwise (no extra particles)*: Evenly distribute the weight sum to all particles.

Recursion: Perform the following steps M times (to get M particles) for each following utterance:

– Sample a particle P based on particle weights (that indicate relative confidence about particle positions);

– Generate a next particle new_P by:

- (i) computing a radius (R) estimated by assumed vehicle speed and the elapsed time from the last particle computation (i.e., the time when the last user utterance was given);

- (ii) selecting a new location (loc) randomly within the radius R from the selected particle P (currently, we do not extract directional information, if any, from utterances that can guide which direction to focus on). Note that it is

also possible to select a new particle location based on a probability distribution (e.g., Gaussian) that models vehicle movement;

- (iii) selecting new_P's final location by computing the closest lat/lon coordinates corresponding to a point on the road network (within the radius R) from the location loc computed in (ii).

After getting M particles, we then compute new weights for them. Weights are inversely proportional to the distance between particles and the potential referents (search results) for the geospatial references mentioned in user utterances. In other words, the closer a particle to a search results, the higher its weight. The weights are then normalized before being assigned to the new particles.

6.1.2. *Path Prediction*

There are several ways to convert the particle filter model into an actual location prediction. The most straightforward approach is to predict the location of the particle with the highest weight. We found better performance when using a centroid-based prediction method. In this approach, we sort the particles by weight (in descending order) and then take the top *N*th percentile of the list by weight. The predicted location is then the weighted centroid of that group of particles. There are, however, cases (typically at the start of a session) in which the particles are spread out, which can result in the centroid being quite far from any given particle. To prevent this situation, if the predicted location is more than a certain distance from the highest-weighted particle (1 km in our evaluation), we default to predicting the location of the highest-weighted particle instead.

6.2. *Experimental Results*

We used two sessions (three audio tracks) for development: tuning parameters, deciding on the path prediction model, etc. The other five sessions (ten audio tracks) were used for testing.

The text-based system was run separately on each of the ten test tracks. The system was independent of any other training data, except the named entity recognizer (NER), which was trained on the set of all sessions except the current route (as was done in the evaluation of the *post hoc* system). The system had the option of making a location prediction (lat/lon) after each utterance from the test set. However, since the data was gathered with an open microphone, it includes a number of utterances that are extraneous to making location predictions (such as comments on the aesthetic value of buildings). These were also fed to the system, but as they contain little or no pertinent information value, these were not treated as prediction opportunities in the evaluation. The prediction was based on the centroid of the top 50 percentile of the highest weighted particles in a simple particle filter (using 100 particles). Performance was averaged over ten runs of the system. Location predictions were counted as correct if they were within 300 m of the actual position.

We evaluated the system's performance using micro-averaged precision and recall. The system itself made or did not make predictions based upon its model and whether it had extracted any location reference from the utterance. Thus, there were utterances for which it made no new update to its model and thus offered no new location prediction. These were counted as false negatives.

Additionally, we measured the distance between the predictions and the actual location, to give a sense of how close to the original location the system is predicting.

On the test data, the text-based system got 80.7% precision and 62.1% recall (70.2% f-score). For correct predictions, the average distance to the actual location was 96 m. Overall, the average distance between the system's predictions and the true path was 230 m.

6.3. *Discussion*

Precision for the online version of TEGUS (80.7%) was much lower than that for the *post hoc* version (92.6%), but recall was significantly higher (62.1% versus 49.2%) as was the f-score (70.2% versus 64.3%). Of note, the online prediction has the added benefit of potentially being useful to many more applications (as we will discuss in more detail below).

We believe this performance can be greatly improved by both improving language understanding and modeling the path prediction based on street network movement. Right now, the system is only extracting references to geospatial entities and searching for them in a geospatial database. This is actually ignoring much of the rich language that could contribute to prediction location, such as references to spatial relations, orientation, and movement events. We discuss this issue more below in future work.

On average, our correct predictions are 96 m from the actual location. Although good, we believe this can also be improved by implementing some of the ideas discussed above. Note also that we put very little effort into the particle filter used to model the evidence, as this was not the research focus of this project. We believe that a more sophisticated localization mechanism will also improve performance.

7. Dialog-Based Geolocation

In this section, we describe the final version of TEGUS, which uses dialog to augment the online path prediction system described in the previous section. The previously described versions of TEGUS made path location predictions based only on the input utterances. In the dialog-based system, however, the system can ask clarification questions to better help narrow down location predictions.

7.1. *System Description*

This version of TEGUS augments the online version with the following features (which we will describe in turn). First is the addition of speech recognition to allow speech input. [5] Second is the addition of a simple dialog manager that generates questions for the user. Last is the incorporation of questions and answers into the particle filter for path prediction.

7.1.1. *Speech Recognition*

For a dialog system, one of the important issues is the development of language models (LMs) for the speech recognition component. Typically, LMs are trained using existing data (corpora) for the task at hand; when no such corpora exist, they need to be collected first — a lengthy and costly endeavor. In the past we found that for dialog systems it is the norm rather than the exception that adequate corpora cannot be found. We thus developed techniques for the fast development of broad-vocabulary LMs from very small corpora of in-domain utterances, by generalizing these corpora into finite state grammars which are then used to generate much larger artificial corpora (Galescu *et al.*, 1998). In this project, we expanded our previous methodology by adding a couple of innovative adaptive features. First, in addition to a small set of dialog-style set of utterances for developing the finite state grammar, we use the development set of PURSUIT Corpus as a source of language data; its descriptive style is not a perfect match for the interactive style used in dialog, but other than that it's a fairly good approximation. However, rather than using it just for training word-based LMs (it is a small amount of data, after all), we map the annotated location categories into non-terminals in the finite state grammar, thereby allowing us to achieve a sort of cross fertilization between the finite state grammar approach and the corpus-based approach. Second, we generate class LMs, where certain classes of words/phrases (e.g., place names and street names) are not expanded into words. At run-time we query for named entities in the area of interest and use those for class expansion to transform the class LM into a word LM suitable for speech recognition. This process will result in the computation of a new LM after each location prediction. We expect these adaptive LMs are significantly higher quality than static LMs that include all named entities in the larger area of interest, although empirically testing this remains an effort for future work.

7.1.2. *Dialog Management and Question Generation*

In this version of TEGUS, we included a fairly rudimentary dialog manager for question generation and processing. After each user utterance describing a location, the system internally makes an initial location prediction using the same particle filter algorithm as the online version of the system described above. Based on this predic-

5. Although the previous versions of the system could theoretically take speech input, we only included it for the dialog system to help in demonstrations. We mention it here, though, as it is an area that deserves more attention in terms of processing geospatial language.

tion, the system has the option to generate a question about the user's location; if the user answers this question, the system uses the additional information in the answer to make a new prediction, which becomes the final hypothesis for the location referenced in the initial user utterance.

In the system, we use a very basic algorithm to generate questions (generating and using more sophisticated questions is an area of future work). The system takes the lat/lon coordinates of the top prediction and queries the geospatial database for entities nearby that point. If an entity is close by, and it hasn't been mentioned before, the system then asks the user if he is "near" the given entity. This sets up a dialog context in which the system waits for a "yes" or "no" answer. The user can also ignore the question and give another, unrelated description utterance, in which case the system proceeds as if no clarification question had been asked (the initial location hypothesis becoming final). The dialog manager also keeps track of each geospatial entity that has been mentioned in the dialog (either by the system or the user) and does not generate questions about these.

7.1.3. *Path Prediction*

If the user's answer to the system's question is "yes", the system treats this as if the user had uttered "I am near X" (where X is the geospatial entity in the clarification question). This information is then used to update the particle filter and make a new prediction as described in the previous section.

The case where the user's answer is "no" is a bit more complicated. First, it is important to consider what a "no" answer implies. It could mean the user is not near the entity, or it could also mean the user is not within visual view of the entity and yet still near it. Or, it could mean that the user simply did not notice the entity nearby. Because of this uncertainty, we decided not to treat a negative reply as absolute evidence (and, for example, make the probability of that area 0).

Instead, with negative responses, we adjust particle weights as follows (the locations of the particles do *not* change):

– (i) Select particles that are located within a certain distance from query results from extracted geospatial references in the preceding utterance;

– (ii) Decrease weights of those particles by some percentage;

– (iii) With the amount of weights decreased at (ii), distribute it to other particles that are not selected at (i), if any.

The updated model (with adjusted weights) is then used to make a new location prediction, as described above.

7.2. *Experimental Results*

To evaluate the extent to which dialog helps improve the system's performance, we devised a method to test the dialog system on the same data set, and under the same testing protocol as in the previous section. Specifically, we created a component to simulate user responses to the questions. This *simulated user*, when asked by the system if some geospatial entity was nearby, checked the actual location of car (from the PURSUIT annotations) at that time, and, if it was within 200 m, it would answer "yes"; otherwise, it would answer "no". In this case, the simulated user is always correct in its "no" answers, but we still process them cautiously, as outlined above.

Locations are predicted as before, for each utterance of the test set, but this time the system's final prediction is made after the system asks its question and evaluates the answer. Performance was, again, averaged over ten system runs.

On this test, the dialog system obtained 86.5% precision and 63.5% recall, for a combined f-score of 73.3%. For correct predictions, the average distance to the actual location was 94 m. Overall, the average distance between the system's predictions and the true path was 176 m.

7.3. *Discussion*

As expected, moving to a question-and-answer dialog did improve system performance. Precision benefited most, with a boost of over 7% (86.5%, compared to 80.7% for the non-interactive system). Recall also improved by a more modest 2.5% (63.5%, compared to 62.1%). As a result, the f-score increased by almost 4.5%. It appears that the additional questions did help the system "nudge" its predictions closer to the ground truth. This is revealed most dramatically by the over 23% reduction in the average distance between the system's predictions and the ground truth (from 230 m to 176 m). Correct predictions only got an insignificant improvement (from 96 m to 94 m), though, of course, the set of correct predictions is larger for the dialog system than for the non-interactive system.

We further wanted to know whether our treatment of negative answers to the system's questions was, indeed, helpful. We therefore ran another ten system runs on the same data, but this time we had the system ignore all negative answers to its questions, and act only on positive answers. This case effectively tests also the case where the user would answer "I don't know" instead of "no", which is, in fact, quite plausible for the situation where the user doesn't see the entity in the system's question. For this test, we obtained 83.6% precision, 62.8% recall (f-score was 71.7%); the average distance to the correct locations was 271 m for all predictions and 104 m for just the correct ones. Thus, it appears that negative answers contribute about the same as positive answers towards the performance improvements obtained by the dialog system. However, the negative answers seem to have a much larger effect on how close the path predicted by the system stays to the ground truth.

We note here that, while the simulated user technique allowed us to directly compare the results of the interactive system with its non-interactive counterpart, it has one limitation, in that it will always answer "yes" when it knows the entity in the system's question is close to the true location. In reality, for a real user-in-the-street scenario, it is likely that the user may not see some landmark even if it is nearby, because of obstructions, improper signage, and so on. As mentioned above, a user may ignore the system's question (or just answer "I don't know"), but we don't have a model of how often this might happen in reality. Thus, while the results reported above unequivocally show that system-generated questions are beneficial, they probably overstate to some extent their usefulness.

On the other hand, there are obvious areas where the dialog system could do a better job and thus further improve its performance. The current system uses a very basic strategy for generating clarification questions — looking up the nearest entity to the prediction and asking if the user is "near" it. There are two areas in which this can be vastly improved. First, the system needs the ability to generate a location to ask about using some sort of measure of expected utility of the answer. Second, the system can be expanded to use the more complicated language mechanisms to ask a more targeted question (such as "Are you across the street from X?"). Such questions can focus the range of the question as well as potentially direct the user's attention to the given place, to increase the chances of the user correctly knowing whether the entity in question is actually there.

8. Conclusion and Future Work

In this paper, we have described our work on predicting geospatial path locations at a street level from natural language descriptions. We described the PURSUIT Corpus, which contains path descriptions coupled with GPS tracks for routes driven in an urban area. Geospatial entity references in the corpus have been annotated with referents from geospatial databases and the corpus is freely available for other researchers.

We then described three versions of a path geolocation system we built (*post hoc*, online, and dialog-based), and their evaluations using the PURSUIT Corpus. We showed that system questions can help improve performance on the geolocation task.

There are many possible future directions to this work. First off, we believe that one of the reasons for the good performance of our system is that, in addition to the language descriptions, it uses temporal information about elapsed time between utterances, which allows it to discard impossible transitions based on that time and an assumed speed. However, timing information is not always available, especially for processing textual information. We would like to explore ways to generalize this away from time, as well as away from paths, to be able to geolocate any and all geospatial references within text.

The current system, in all the three modes, is based only on the processing of geospatial entities, and does not yet utilize the rich descriptive power of natural lan-

guage. Although it performs quite well, this is based on the simplifying assumption that each place mentioned by the user is "near" to the user. Using better language understanding (and constraint modeling based on it) should result in much more accurate predictive power for the system. There are several areas of geospatial language understanding that need to be taken into account:

Relations In addition to a generic "near" relation, natural languages encode a number of spatial relations, and many of these appear in the PURSUIT Corpus. Such relations are often (but not always) expressed as prepositions in English and encode specific spatial relationships between entities. These include both quantitative (e.g., "100 meters away") and qualitative relations. Of the latter, examples include containment (e.g., "inside", "outside"), position (e.g., "in front of", "behind", "to the left of", "to the right of"), and many others. Each of these can be modeled as a constraint on the relative position of the two entities. Spatial relations have been studied extensively in Linguistics (e.g., Logan and Sadler, 1996; Levit and Roy, 2007). The research challenge here is extracting the relations automatically, and modeling them as sensor output (most likely a probabilistic distribution over a 2-D shape on the map);

Orientation Language also includes descriptions of spatial orientation, and the PURSUIT Corpus includes instances that describe the orientation of the user's movement, as well as the orientation of spatial entities (such as streets or buildings). Examples include both extrinsic orientation (e.g., "north", "south") and intrinsic orientation (e.g., "turning left"). These also can be modeled as constraints on position and movement;

Movement Events In cases where movement is present, the bulk of the language we have encountered describes movement events, such as "turning <direction>", "stopping", "passing <geospatial entity>". Research is needed to catalog the set of these and build information extraction rules to extract them from language. We also need tools to model movement on a street (or footpath, etc.) network and reason about constraints over it. Note that, with events, not only the event type, but also its parameters need to be extracted;

Spatial Regions Finally, we mention here that the current system does not currently handle spatial regions (such as neighborhoods, cities, states, etc.) Instead, all entities are treated as points. On a sub-city scale, we need to be able to extract mentions of regions and also reason about location based on these (such as being "in a neighborhood"). Additionally, the current system makes the assumption that an initial bounding box (roughly corresponding to a city) containing the user's location is known. A simple addition to the system would be to allow utterances such as "I'm in Pensacola" to set such bounding boxes and narrow down the scope of search.

Acknowledgements

We would like to thank the reviewers for their helpful comments. We would also like to thank Bradley Swain and Dawn Miller, who helped with the early implementation and annotation work. This work was partially funded under contract #FA8650-11-C-7141 from DARPA. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the US Government.

9. References

Allen J. F., Swift M., De Beaumont W., "Deep Semantic Analysis of Text", *Symposium on Semantics in Systems for Text Processing (STEP 2008) Shared Task: Comparing Semantic Representations*, Venice, Italy, September 22-24, 2008.

Anderson A., Bader M., Bard E., Boyle E., Doherty G. M., Garrod S., Isard S., Kowtko J., McAllister J., Miller J., Sotillo C., Thompson H. S., Weinert R., "The HRCR Map Task Corpus", *Language and Speech*, vol. 34, p. 351-366, 1991.

Barras C., Geoffrois E., Wu Z., Liberman M., "Transcriber: Development and Use of a Tool for Assisting Speech Corpora Production", *Speech Communication special issue on Speech Annotation and Corpus Tools*, January, 2000.

Blaylock N., "Semantic Annotation of Street-Level Geospatial Entities", *Proceedings of the IEEE-ISCS Workshop on Semantic Annotation for Computational Linguistic Resources*, Palo Alto, California, September, 2011.

Blaylock N., Allen J., "Real-Time Path Descriptions Grounded with GPS Tracks: a Preliminary Report", *LREC Workshop on Methodologies and Resources for Processing Spatial Language*, Marrakech, Morocco, p. 25-27, May 31, 2008.

Blaylock N., Allen J., De Beaumont W., Jung H., "Towards an OWL-Based Framework for Extracting Information from Clinical Texts", *Proceedings of the First International Workshop on Semantic Applied Technologies on Biomedical Informatics (SATBI)*, Chicago, Illinois, August 1-3, 2011.

Blaylock N., Swain B., Allen J., "TESLA: a Tool for Annotating Geospatial Language Corpora", *Proceedings North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL HLT) 2009 Conference*, Boulder, Colorado, May 31-June 5, 2009.

Bugmann G., Klein E., Lauria S., Kyriacou T., "Corpus-Based Robotics: a Route Instruction Example", *Proceedings of Intelligent Autonomous Systems (IAS-8)*, Amsterdam, p. 96-103, March 10–13, 2004.

Carletta J., Evert S., Heid U., Kilgour J., "The NITE XML Toolkit: Data Model and Query Language", *Language Resources and Evaluation Journal*, vol. 39, n° 4, p. 313-334, 2005.

Dijkstra E. W., "A Note on Two Problems in Connection with Graphs", *Numerische Mathematik*, vol. 1, p. 269-271, 1959.

Felipe I. D., Hristidis V., Rishe N., "Keyword Search on Spatial Databases", *Proceedings of the 24th International Conference on Data Engineering, ICDE 2008*, Cancun, Mexico, p. 656-665, April 7-12, 2008.

Finkel J. R., Grenager T., Manning C., "Incorporating Non-Local Information Into Information Extraction Systems by Gibbs Sampling", *Proceedings of ACL*, Ann Arbor, Michigan, June, 2005.

Galescu L., Ringger E., Allen J., "Rapid Language Model Development for New Task Domains", *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC)*, Granada, Spain, May, 1998.

Jung H., Allen J., Galescu L., Chambers N., Swift M., Taysom W., "Utilizing Natural Language for One-Shot Task Learning", *Journal of Language and Computation*, vol. 18, n° 3, p. 475-493, 2008.

Knoblock C. A., Chen C.-C., Chiang Y.-Y., Goel A., Michelson M., Shahabi C., "A General Approach to Discovering, Registering, and Extracting Features from Raster Maps", *Proceedings of the Conference on Document Recognition and Retrieval XVII*, 2010.

Leidner J. L., Sinclair G., Webber B., "Grounding Spatial Named Entities for Information Extraction and Question Answering", *Proceedings of the HLT-NAACL 2003 Workshop on Analysis of Geographic References*, Edmonton, Alberta, 2003.

Levit M., Roy D., "Interpretation of Spatial Language in a Map Navigation Task", *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, n° 3, p. 667-679, 2007.

Logan G. D., Sadler D. D., "A Computational Analysis of the Apprehension of Spatial Relations", *in* P. Bloom, M. A. Peterson, L. Nadel, M. F. Garrett (eds), *Language and Space*, MIT Press, Cambridge, Massachusetts, 1996.

MacMahon M., Stankiewicz B., Kuipers B., "Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions", *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, AAAI Press, Boston, Massachusetts, p. 1475-1482, July, 2006.

Mani I., Hitzeman J., Richer J., Harris D., Quimby R., Wellner B., "SpatialML: Annotation Scheme, Corpora, and Tools", *6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, May, 2008.

Manshadi M. H., Allen J. F., Swift M., "Toward a Universal Underspecified Semantic Representation", *13th Conference on Formal Grammar (FG 2008)*, Hamburg, Germany, August 9-10, 2008.

Mikheev A., Moens M., Grover C., "Named Entity Recognition Without Gazetteers", *Proceedings of EACL '99*, Bergen, Norway, p. 1-8, 1999.

O'Connor M., Das A., "SQWRL: a Query Language for OWL", *OWL: Experiences and Directions (OWLED), 5th International Workshop*, Chantilly, Virginia, 2009.

Pustejovsky J., Moszkowicz J. L., "Integrating Motion Predicate Classes with Spatial and Temporal Annotations", *Proceedings of COLING 2008*, Manchester, UK, 2008.

Pustejovsky J., Moszkowicz J. L., Verhagen M., "ISO-Space: the Annotation of Spatial Information in Language", *Proceedings of ISA-6: ACL-ISO International Workshop on Semantic Annotation*, Oxford, England, January, 2011.

Rishe N., Gutierrez M., Selivonenko A., Graham S., "TerraFly: a Tool for Visualizing and Dispensing Geospatial Data", *Imaging Notes*, vol. 20, n° 2, p. 22-23, 2005.

Schmill M. D., Oates T., "Managing Uncertainty in Text-To-Sketch Tracking Problems", *IEEE 23rd International Conference on Tools With Artificial Intelligence*, Boca Raton, Florida, November, 2011.

Sledge I., Keller J. M., "Mapping Natural Language to Imagery: Placing Objects Intelligently", *IEEE Conference on Fuzzy Systems*, Jeju Island, Korea, August, 2009.

Zhang X., Mitra P., Klippel A., MacEachren A. M., "Automatic Extraction of Destinations, Origins and Route Parts from Human Generated Route Directions", *GIScience'10*, p. 279-294, 2010.