A Visual Decision-Guided Analytics Tool for Finding the Viable Shortest Path over Geospatiotemporal Data

Chun-Kit Ngan
Engineering and Information Science Division
Great Valley School of Graduate Professional Studies
The Pennsylvania State University
Malvern, PA, USA
cxn20@psu.edu

Abstract—We propose a Visual Decision-Guided Analytics tool that combines quantitative analysis and qualitative methodologies to determine the viable shortest path over geospatiotemporal data for rescue and recovery missions. Specifically, we first extend the object-oriented spatial-temporal data model as a multidimensional OLAP cube that enables military operators to analyze unified geo-data objects from multiple dimensions, such as time, space, and location, to help them make a better decision on routes. Second, we enhance the capability of the PostGIS query which allows the operators to (1) simulate the occurrence of events, (2) visualize and report geo-data objects, and (3) solve decision optimization problems based upon their events of interest. Third, we integrate optimization programming into geo-data visualization to determine the viable shortest path for rescue and recovery missions. This integration enables military units to make a visual decision on routes based upon the time-distance optimality. Finally, we develop a visual display that enables the operators to analyze other crucial factors, e.g., vehicle types, weather severity, soldiers’ specialties, etc., which are required to be interpreted by human perception, cognition, and knowledge to select the best path among all the viable routes for rescue and recovery missions.

Keywords—decision-support; visual-analytics; optimization; query; OLAP-model

I. INTRODUCTION

In the past decade, military units in nations successfully analyzed integrated geo-data objects to execute decision-making operations. Consider a range of peace-keeping operations that nations’ military forces need to accomplish in order to maintain peace in their countries. These operations include large scale protection operations, peace enforcement, nation assistance, and freedom of navigation [1]. To accomplish these operations, it is important for each military unit, such as Army, Navy, Marine, and Air Force, to cooperate with one another to analyze unified geo-data objects, e.g., satellite imagery, vector/raster maps, temporal data, 3D objects, etc., to support optimal decision-making based upon different unit forces’ data sources. One critical operation that military troops need to determine is to find the viable shortest path to reach a target location for rescue and recovery missions so that medical and health services can be efficiently delivered to victims who can receive supports and supplies in the aftermath of natural disasters. To support such a decision-making operation, it is important for military researchers to develop a geo-data analytical methodology that is reliable and useful for the operation. The main challenge in such a methodology is how to expressively and effectively model and analyze those unified geo-data objects such that diverse military units can analytically make a concerted decision together. This is exactly the focus of this paper.

To support such a decision-making process, a number of researchers has proposed and developed different approaches to model geo-data objects, which can be roughly divided into two categories: On-Line Transaction Processing (OLTP) and On-Line Analytical Processing (OLAP). The former OLTP approach mainly focuses on facilitating and managing transaction-oriented applications that are used for data entry and retrieval processing. For example, Li and Cai [2] proposed an OLTP-based object-oriented spatial temporal data model, from which geo-data objects can be described by a theme, space, and time. Although this data model is efficient for a fast transactional processing analysis, it is not designed for supporting decision-making analytics. The latter OLAP approach enables military operators to analyze multidimensional data interactively from multiple perspectives. For instance, Vaisman and Zimanyi [3] proposed a MultiDim model based on a Spatial On-Line Analytical Processing (SOLAP) cube that can enable military operators to use multi-dimensional views to analyze geo-data objects, from which the operators can gain a more complete decision-making insight. However, since this spatial SOLAP solution does not model real-world objects’ properties and operations based on the object-oriented paradigm, it is hardly suitable for modeling objects’ interactions.

In addition to the data modeling problem, an analytical approach on geo-data objects is another important issue that we need to address. Currently, a number of data analytics approaches has been proposed and developed to analyze geo-data objects. Data mining techniques [4, 5] are computational algorithms that analyze raw data over a terrain to extract valuable information. Some algorithms include rule mining (e.g., the Apriori algorithm) [6], dimensionality reduction methods (e.g., principal components analysis) [7], supervised learning (e.g., decision trees, support vector machines, neural networks, etc.) [8], and unsupervised learning (e.g., cluster analysis) [9]. However, these mining techniques often require expert human interpretation and supervision on data to deliver the results, which are not suitable for a quick decision-making operation. To support military troops to effectively make a better decision, i.e., determining the viable shortest path for
rescue and recovery missions, researchers proposed and developed data visualization tools that expressively display the learned results from the mined geo-data objects to support the analysis. This technique is called visual analytics or visual data mining [5, 10] that combines the advantages of both visualization and data mining methodology.

However, due to the high volume and the large size of the available data, the visual analytics approach is still very cumbersome to be used and can even overwhelm military operators during their decision-making operations. To solve this heavy data problem, a number of operation research methods [11], i.e., Mathematical Programming, e.g., Mixed Integer Linear Programming, Mixed Integer Nonlinear Programming, etc., are revised and streamlined to play a key role in this endeavor. The operation research methods help military units formulate precise objectives, e.g., minimize the cost of traveling time and distance, as well as specific constraints, e.g., the vehicles’ speed limitations, imposed on the solution [12]. Once optimal values of decision parameters (e.g., select/unselect a node and/or a route segment along a path) are learned by optimization algorithms, such as Dijkstra’s [13], Simplex [11], and Branch-and-bound [11], military decision makers can use the learned parameters to determine the shortest path.

Recently, a number of more advanced shortest path learning algorithms has been proposed and developed. The Smoothest Path Algorithm (SPA) [14] is to find the shortest surface path over a terrain in terms of distance and slope rather than the Euclidean distance only. The Shortest Path Algorithm for a Fixed Start Time [15] is to compute the shortest path either for a given start time or to search the start time and the path that results in the least travelling time. However, these approaches do not consider taking advantages of one another to determine the shortest route based upon the distance, slope, time schedule, and other possible terrain obstacles together. In addition, the learned paths computed by those algorithms only reflect an optimal reality under some environmental constraints rather than address the entire phenomenon, especially after natural disasters. For instance, the shortest path delivered by those models and algorithms are not considered to include other crucial factors, such as vehicle types, weather severity, moving entities, and soldiers’ specialties, during the learning computation. These neglected factors definitely impact the traveling time and distance that still require human perception, cognition, and knowledge for making a final decision.

Thus this paper focuses on bridging the above research gaps among (1) the geo-data modeling, (2) the viable shortest path determination, and (3) the visual analytics. Specifically, we propose a Visual Decision-Guided Analytics (VEGA) tool that is a unified decision-making application that combines both advantages of quantitative analysis (i.e., optimization programming in operation research) and qualitative methodologies (i.e., geo-data visualization in data analytics) to determine the viable shortest path for rescue and recovery missions. This application supports reporting, simulation, visualization, and decision optimization [16] over geospatiotemporal data. Technically, this tool enables military units to (1) collect geo-data objects from multiple data sources, (2) conflate the diverse objects into unified data sets, and (3) perform analysis on these sets. First, to support a better decision-making, we extend the object-oriented spatiotemporal data model as a multidimensional OLAP (star-schema) cube, i.e., a Star-based Geo-Object-Oriented SpatioTemporal (S-GOOSE) data model, which combines the advantages of both OLTP and OLAP approaches. This S-GOOSE data model is an object-relational-based cube that enables military operators to analyze unified geo-data objects from multiple dimensions, such as time, space, and location, to help them make a better decision on routes. Second, we enhance the capability of the PostGIS query [17], named Geo-Query Language (GQL), to support the S-GOOSE data cube analysis, which enables military operators to simulate the occurrence of events, to visualize and report geo-data objects, as well as to solve decision optimization problems based upon their events of interest. The reason to select the PostGIS query to be extended is because the GIS objects supported by the PostGIS query are a superset of the "Simple Features" defined by the OpenGIS Consortium (OGC) [18]. The PostGIS query supports all the objects and functions specified in the OGC "Simple Features for SQL" specification [17]. Third, we integrate optimization programming into geo-data visualization to determine the viable shortest path for rescue and recovery missions. The idea is to enable military units to make a visual decision on routes based upon the time-distance optimality among all the possible paths. Specifically, we are developing the Top-k Objected-oriented Smoothest Paths (TOSP) model which captures the object dynamics of geospatial temporal network in a terrain over a time horizon. These objects include stationary entities (e.g., buildings, roads, trees, etc.), mobile objects (e.g., vehicles, people, etc.), and route segments (e.g., steep slopes, mud roads, etc.). We are also extending the SPA to be a dynamic learning algorithm, i.e., the Time-varying Smoothest Path (TSP) algorithm, which integrates the object dynamics to learn the top-k smoothest paths at each instance of time. The main advantage offered by the SPA extension is its lower logarithmic time complexity, i.e., $O(N \log N)$, where N is the number of nodes in a terrain. Finally, we develop a new design of visual displays that enable military operators to analyze other crucial factors, such as vehicle types, weather severity, and soldiers’ specialties, which are required to be interpreted by human perception, cognition, and knowledge to select the best path among the top-k smoothest routes at each instance of time for rescue and recovery missions.

The rest of the paper is organized as follows. In Section II, we use the above military operation, i.e., determining the viable shortest path for the rescue and recovery mission, to provide an overview on our VEGA Tool. In Section III, we describe the S-GOOSE data model and GQL, as well as illustrate the model and language based upon the military case. In Section IV, we present the implementation architecture for the viable shortest-path learning process. In Section V, we explain the initial design of our high-level 3D visual display for determining the viable shortest path among the top-k smoothest routes. In Section VI, we conclude our paper and briefly outline our future work.
II. VISUAL DECISION-GUIDED ANALYTICS TOOL

Fig. 1 shows the VEGA tool and its components. The VEGA tool consists of seven main components: Geo-Data Source Collector (GDSC), Geo-Data Fusion Integrator (GDFI), Geo-Data Warehouse (GDW), Geo-Query Language (GQL), Data Visualizer (DV), TOSP Compiler, and TSP Solver. The GDSC allows military operators to directly interact with heterogeneous data sources, e.g., satellite imagery, vector/raster maps, temporal data, 3D objects, etc., and gather those geo-data objects from multiple units, including Army, Navy, Marine, and Air Force.
After geo-data objects are collected, military forces can operate the GDWI to clean the objects and then integrate them into a S-GOOSE data cube, which provides and maintains a concentric and coherent view of the collected data. These cubes are then archived in the GDW, that is, the extended S-GOOSE data warehouse. The unified cubes can also be decomposed into different terrain layers according to their themes, e.g., trees, buildings, mountains, roads, etc., which are also stored in the GDW as well for future analysis.

The GQL enables military operators to (1) develop and implement the extended S-GOOSE data cubes and (2) construct different views, including reporting, simulation, visualization, and decision optimization, based on the integrated geo-data objects to support determining the viable shortest path for rescue and recovery missions. More specifically, the GQL enables military operators to (1) retrieve data from the GDW and (2) construct reporting (RV), simulation (SV), visualization (VV), and decision optimization (DOV) views based on the unified geo-data objects. Once military operators initiate the data and DOV query construct to the TOSP compiler, the query translator transforms the DOV query into the TOSP format, which is then sent to the TSP solver with the TSP algorithm to learn the top-k smoothest paths at each instance of time. After the compiler receives the top-k route segments and objectives, the output formatter renders the results as the optimized answers to the DOV query. In order to provide the future insight, military operators can also construct the SB query to simulate the top-k smoothest paths at the next instance of time. The answers from both DOV and SB query, as well as the visual form from the VV query are then integrated by the $\oplus$ aggregator, which delivers the aggregated results to the DV. If it is needed, military operators can also formulate the RV and VV query for constructing a report, e.g., the total number of buildings in each city per state at a specific time after a natural disaster.

The DV displays analytical diagrams and figures, e.g., 2D pie charts and maps, 3D bar graphs and maps, etc., to military operators. Military operators, e.g., the ground soldiers, can use the pull or push services provided by the mobile network and devices to receive the latest visual information about the routes. The military command center can also base on the visual information to guide the soldiers’ next movements via the communication network. Collaboratively, using our designed VEGA visuals, both parties are able to make a final decision on the viable shortest path based on the optimal and simulated results of the top-k smoothest paths, as well as the other crucial factors, e.g., vehicle types, weather severity, and soldiers’ specialties.

III. S-GOOSE DATA MODEL AND GEO-QUERY LANGUAGE BY EXAMPLE

In this section, we discuss in detail the S-GOOSE data model and GQL, which are used for reporting, simulation, visualization, and decision optimization. Again, we use the military operation, i.e., determining the viable shortest path for rescue and recovery missions, to illustrate the ideas.

A. S-GOOSE Data Model

The S-GOOSE data model is an extension of the object-relational database model with a specialized schema, i.e., a S-GOOSE OLAP cube with a number of dimension schemas. The dimension schemas include Time, Location, Visualization, and Composite Object in a terrain over a time horizon. To demonstrate the concepts, we assume that there is a building layer and a road layer in a disaster region and the military operational unit passes through a sequence of buildings along some road segments in an area to rescue a group of victims, who are trapped inside the target building. The extended S-GOOSE OLAP cubes of the building and the road layers are shown in Fig. 2 and 3 respectively.

![Fig. 2. S-GOOSE OLAP Cube for the Building Layer.](image1)

![Fig. 3. S-GOOSE OLAP Cube for the Road Layer.](image2)
**Time Dimension**

A time dimension is of the form Time(id:INTEGER, yyyy:YEAR, mo:MONTH, dd:DAY, hh:HOUR, mm:MINUTE, ss:SECOND), where YEAR, MONTH, DAY, HOUR, MINUTE, and SECOND are the data types of yyyy, mo, dd, hh, mm, and ss respectively. Each data type is an object-relational table that has an attribute id to define a property of that type. For example, the attribute id '12' in the MONTH table represents December. The attribute id '15' in the MINUTE table represents the first quarter of an hour, i.e., 15 min.

A tuple over a Time schema is an object-relational tuple over that schema, i.e., a mapping m: {id, yyyy, mo, dd, hh, mm, ss} → INTEGER x YEAR x MONTH x DAY x HOUR x MINUTE x SECOND, such that m(id) ∈ INTEGER, m(yyyy) ∈ YEAR, m(mo) ∈ MONTH, m(dd) ∈ DAY, m(hh) ∈ HOUR, m(mm) ∈ MINUTE, and m(ss) ∈ SECOND.

**Location Dimension**

A location dimension is of the form Location(id:INTEGER, long:DOUBLE, lat:DOUBLE, state:STATE, zipcode:ZIP, city:CITY, street:STREET), where STATE, ZIP, CITY, and STREET are data types of state, zipcode, city, and street respectively. Each above data type is an object-relational table that has an attribute id to define a property of that type. For example, the attribute id '2032' in the ZIP table represents a zipcode of the city. The attribute id 'VA' in the STATE table represents a state, i.e., Virginia. Note that long and lat are the longitude and the latitude of an object on the map.

A tuple over a Location schema is an object-relational tuple over that schema, i.e., a mapping m: {id, long, lat, state, zipcode, city, street} → INTEGER x DOUBLE x DOUBLE x STATE x ZIP x CITY x STREET, such that m(id) ∈ INTEGER, m(long) ∈ DOUBLE, m(lat) ∈ DOUBLE, m(state) ∈ STATE, m(zipcode) ∈ ZIP, m(city) ∈ CITY, and m(street) ∈ STREET.

**Visualization Dimension**

A visualization dimension has four layers of schemas: 3D(id:INTEGER, surface:SURFACE[]), Surface(id:INTEGER, line:LINE[]), Line(id:INTEGER, point:POINT[]), and Point(id:INTEGER, xc:DOUBLE, yc:DOUBLE, zc:DOUBLE), where SURFACE[], LINE[], and POINT[] are the array types of surface, line, and point attributes respectively. Each above data type is an object-relational array that stores a set of its component objects to construct a composite object. For example, the array attribute 'surface' in the 3D table stores all the ids of the surfaces defined in the Surface table to construct a 3D object. The array attribute 'line' in the Surface table stores all the ids of the lines defined in the Line table to construct a 2D object. The (xc, yc, zc) in the Point schema is the actual coordinate (longitude, latitude, elevation) on a contour line of an object, e.g., a tree, a house, a road, a building, etc.

A tuple over a 3D schema is an object-relational tuple over that schema, i.e., a mapping m: {id, surface} → INTEGER x SURFACE[], such that m(id) ∈ INTEGER and m(surface) ∈ SURFACE[].

A tuple over a Surface schema is an object-relational tuple over that schema, i.e., a mapping m: {id, line} → INTEGER x LINE[], such that m(id) ∈ INTEGER and m(line) ∈ LINE[].

A tuple over a Line schema is an object-relational tuple over that schema, i.e., a mapping m: {id, point} → INTEGER x POINT[], such that m(id) ∈ INTEGER and m(point) ∈ POINT[].

A tuple over a Point schema is a relational tuple over that schema, i.e., a mapping m: {id, xc, yc, zc} → INTEGER x DOUBLE x DOUBLE x DOUBLE, such that m(id) ∈ INTEGER, m(xc) ∈ DOUBLE, m(yc) ∈ DOUBLE, and m(zc) ∈ DOUBLE.

**Composite Object Dimension**

A composite object dimension is of the form CompObj(id:INTEGER, [name:STRING], [height:DOUBLE], [width:DOUBLE], [length:DOUBLE], [compObjName:COMPOBJTYPE]). A CompObj is an object that can be constructed by a set of its component objects. The optional attributes, 'height', 'width', and 'length', are the dimension of an object with the optional 'name'. [compObjName:COMPOBJTYPE] is a set of optional object-relational arrays that stores a set of their component objects to construct a composite object. Each component object can be constructed from another set of component objects.

A tuple over a CompObj schema is an object-relational tuple over that schema, i.e., a mapping m: {id, [name], [height], [width], [length], [compObjName]} → INTEGER x [STRING] x [DOUBLE] x [DOUBLE] x [DOUBLE] x [COMPOBJTYPE], such that m(id) ∈ INTEGER, m(name) ∈ STRING, m(height) ∈ DOUBLE, m(width) ∈ DOUBLE, m(length) ∈ DOUBLE, and m(compObjName) ∈ COMPOBJTYPE.

**S-GOOSE Database Schema**

The S-GOOSE database schema is a set of object-relational schemas, which include a number of S-GOOSE OLAP cubes.

Using the extended S-GOOSE OLAP cubes of the building layer and the road layer shown in Fig. 2 and 3 respectively, the military forces can (1) create the S-GOOSE tables and views, including reporting, simulation, visualization, and decision optimization, (2) store the tables and views with the data in the GDW, and (3) execute the views to perform the analysis in different combinations of dimensions.

**B. Reporting View**

Using the Time, Location, and Composite Object dimensions, the military force can generate a report to display...
the total number of buildings in each city, e.g., Fairfax, in a state, e.g., VA, at a specific time, e.g., March 15, 2014, after the natural disaster. This reporting view, shown in Box 1, can be constructed by using the conventional PostGIS query.

Box 1

```
CREATE VIEW BuildingReport AS
(SELECT L.state, L.city, COUNT(DISTINCT B.id) AS "# of Cities"
FROM Time T, Location L, Building B, Fact F
WHERE T.id = F.tid AND L.id = F.lid AND
GROUP BY L.state, L.city
ORDER BY L.state, L.city)
```

C. Decision Optimization View

The decision optimization query helps the military troop compute and learn the top-k smoothest paths to reach the target building at each instance of time for the rescue and recovery mission. The procedures for constructing the view are shown in the following steps:

**STEP 1:** Create a BuildingCoordinate view shown in Box 2 to retrieve all the buildings’ coordinates in terms of their latitudes (lat) and longitudes (long) within the building layer in a particular city, e.g., Fairfax, at a specific time, e.g., March 15, 2014.

Box 2

```
CREATE VIEW BuildingCoordinate AS
(SELECT B.id, L.long, L.lat
FROM Time T, Location L, Building B, Fact F
WHERE T.id = F.tid AND L.id = F.lid AND
)
```

**STEP 2:** Create a RoadCoordinate view shown in Box 3 to retrieve all the road segments’ coordinates in terms of their lat and long within the road layer in the same city, i.e., Fairfax, on March 15, 2014.

Box 3

```
CREATE VIEW RoadCoordinate AS
(SELECT R.id, L.long, L.lat
FROM Time T, Location L, Road R, Fact F
WHERE T.id = F.tid AND L.id = F.lid AND
)
```

**STEP 3:** Create a BuildingRoadMap view shown in Box 4 to retrieve all the buildings’ coordinates within a distance of, e.g., 0.1 mile (5280 feet x 0.1), of their road segment’s coordinate in that area. Note that `ST_DWithin()` [19] and `ST_MakePoint()` [20] are the spatial relationship functions of the PostGIS query [15].

Box 4

```
CREATE VIEW BuildingRoadMap AS
(SELECT RC.long AS rlong, RC.lat AS rlat,
BRM.rlong, BRM.rlat, BRM.blong,
BRM.blat
FROM BuildingCoordinate BC,
RoadCoordinate RC
WHERE
ST_DWithin(ST_MakePoint(BC.long, BC.lat),
ST_MakePoint(RC.long, RC.lat), 5280 * 0.1)
)
```

**STEP 4:** Create a decision optimization view shown in Box 5 and then execute the view `DetermineTopKSmootherPaths` to determine the top-k sequences of all the buildings (BRM.rlong, BRM.rlat) within a distance of 0.1 mile of their road segments within March 15, 2014. Note that the total distance in each sequence between the military force’s present location and the target building location is minimal in order.

Box 5

```
CREATE VIEW DetermineTopKSmootherPaths AS
(LEARN SEQ(BRM.rlong, BRM.rlat, BRM.blong, BRM.blat)
FOR MINIMIZE
ST_TopK_Smoother_Paths(BRM.rlong, BRM.rlat, BRM.blong, BRM.blat,
presentLong, presentLat, targetLong, targetLat)
FROM BuildingRoadMap BRM
)
```

Once the above query construct is EXECUTED and initiated to the GDW, the VEGA tool invokes the Top-k Smoothest Paths Optimizer to LEARN the top-k sequences SEQ of the buildings’ coordinates along their paths on the BRM. Note that `ST_TopK_Smoother_Paths` is a new function being developed, which accepts the four sets of input parameters, including the roads’ coordinates, their buildings’ coordinates, the troop’s present location, and the target building location, to compute and learn the top-k smoothest paths based upon the BRM. The SEQ is another new function to return the top-k sequences of all the buildings’ coordinates along their roads’ coordinates delivered by the `ST_TopK_Smoother_Paths` function.
D. Simulation View

Similarly, following the above STEP 1 - 4, we can also construct the simulation view to predict the top-k smoothest paths at the next instance of time. Some existing methodologies, e.g., random-walk, random-trend, autoregressive, exponential smoothing, etc., can be used to generate those forecasting data based on the real-time dataset to learn the top-k smoothest paths.

E. Visualization View

To support military troops to make a better decision on the viable shortest path for their rescue and recovery missions, various data visualization approaches are proposed and developed to assist military units in performing visual analytics over object-oriented spatial-temporal data. Those techniques present and deliver visual objects that require human interpretation and supervision on data. To support human interpretation and supervision on those visual objects, military units can construct the visualization view, e.g., the distribution view of all the building objects along their paths in a terrain at an instance of time. Box 6 shows an example of a visual query that the military troops can construct to display the building layer in the terrain after a natural disaster.

**Box 6**

```sql
CREATE VIEW 3DBuildingShape AS
(SELECT P.xc, P.yc, P.zc
FROM Point P
WHERE P.id COMPOSE
(SELECT 1D.id
FROM Line 1D
WHERE 1D.id COMPOSE
(SELECT 2D.id
FROM Surface 2D
WHERE 2D.id COMPOSE
3D.surface[3D.id])
FROM Time T, Location L, Building B, 3D,
Fact F
WHERE T.id = F.tid AND L.id = F.lid AND
)
VISUALIZE 3DBuildingShape;
```

Please note that `COMPOSE` is a new keyword to evaluate whether an object constructs another object. For instance, the syntax `P.id COMPOSE 1D.point[1D.id]` means that an actual point coordinate, which is one of the components, constructs a contour line of an object, that is, a building. Likewise, the syntax `2D.id COMPOSE 3D.surface[3D.id]` means that a 2D object, which is one of the components, constructs a 3D object. `VISUALIZE` is another new keyword to execute the `3DBuildingShape` view to display a 3D building layer in a terrain at an instance of time.

IV. IMPLEMENTATION OF A HIGH-LEVEL ARCHITECTURE FOR THE VIABLE SHORTEST-PATH LEARNING PROCESS

Fig. 4 illustrates the viable shortest-path learning process. As this figure shows, the military operators can use the GQL interface to construct the DOV and SV query for the learning event, e.g., `DetermineTopKSmoothestPaths`. Once this learning event aggregated with the S-GOOSE OLAP cube is initiated to the GDW, the TSP optimizer invokes the TOSP compiler, which calls the query translator to transform the learning event into the TOSP construct. This TOSP construct is then sent to the TSP solver to learn the top-k sequences of the buildings’ coordinates along their route segments and objectives. These top-k sequences and objectives are then processed by the output formatter associated with the query translator to return the optimized and/or simulated answers back to the GQL interface, which presents the results to the operators. Finally, the operators can formulate the visualization view aggregated with the optimized and/or simulated answers to be displayed on the screen by the Data Visualizer.

![Fig. 4. The Viable Shortest-Path Learning Architecture.](image)

V. INITIAL DESIGN OF 3D VISUAL DISPLAY

The initial design of our high-level 3D visual display for the optimized and the simulated top-k smoothest paths is shown on the mobile devices as an example (see Fig. 5 and 6 respectively). These visual displays are updated and refreshed for every instance of time. Fig. 5 screen display is divided into two portions. The right-hand portion shows the top-k optimal paths on the map, where the red line is the first optimal path, and the blue line is the second at the current time point t. Both of the paths are learned by our TSP algorithm. The left-hand portion displays the road characteristics, such as grades (e.g., HILL), conditions (e.g., BUMP), and speeds (e.g., 35 MPH), using the standard road signs, as well as shows the current weather (e.g., Sunny), the soldier’s specialties (e.g., Corporal), and their vehicle types (e.g., a Four-wheel Truck). Fig. 6, which has the same visual layout as Fig. 5, shows the top-k simulated paths on the map, where the green line is the first optimal, simulated path, and the pink line is the second at the future time point t + Δt. Due to the dynamics of geospatial temporal network in a terrain over a time horizon, the simulated results render different top-k paths learned by our TSP algorithm. Using the both optimal and simulated paths
VI. Conclusions and Future Work

In this paper, we propose a VEGA tool that is a unified decision-making application that combines both advantages of quantitative analysis (i.e., optimization programming in operation research) and qualitative methodologies (i.e., geodata visualization in data analytics) to determine the viable shortest path for rescue and recovery missions. This application supports reporting, simulation, visualization, and decision optimization over geospatiotemporal data. Technically, this tool enables military units to (1) collect geo-data objects from multiple data sources, (2) conflate the diverse objects into unified data sets, and (3) perform analysis on these sets. First, to support a better decision-making, we extend the object-oriented spatial-temporal data model as a multidimensional OLAP (star-schema) cube, i.e., a Star-based Geo-Object-Oriented Spatiotemporal (S-GOOSE) data model, which combines the advantages of both OLTP and OLAP approaches. This S-GOOSE data model is an object-relational-based cube that enables military operators to analyze unified geo-data objects from multiple dimensions, such as time, space, and location, to help them make a better decision on routes. Second, we enhance the capability of the PostGIS query, named Geo-Query Language (GQL), to support the S-GOOSE data cube analysis, which enables military operators to simulate the occurrence of events, to visualize and report geo-data objects, as well as to solve decision optimization problems based upon their events of interest. Third, we integrate optimization programming into geo-data visualization to determine the viable shortest path for rescue and recovery missions. The idea is to enable military units to make a visual decision on routes based upon the time-distance optimality among all the possible paths. Finally, we develop a new design of visual displays that enable military operators to analyze other crucial factors, such as vehicle types, weather severity, and soldiers’ specialties, which are required to be interpreted by human perception, cognition, and knowledge to select the best path among the top-k smoothest routes at each instance of time for rescue and recovery missions. There are still many open research questions, particularly conflating multiple geo-data objects, e.g., satellite imagery, vector/raster maps, temporal data, 3D objects, etc., into an integrated data unit.

REFERENCES