BRECCIA: A Novel Multi-source Fusion Framework for Dynamic Geospatial Data Analysis

D. Sacharny, T.C. Henderson, R. Simmons, A. Mitiche and T. Welker
School of Computing
University of Utah
Salt Lake City, UT, USA
Email: tch@cs.utah.edu

X. Fan
Nanyang Technological University
Singapore
Email: xyfan@ntu.edu.sg

***Abstract*— Geospatial Intelligence analysis involves the combination of multi-source information expressed in logical form (as sentences or statements), computational form (as numerical models of physics or other processes), and sensor data (as measurements from transducers). Each of these forms has its own way to describe uncertainty or error: e.g., frequency models, algorithmic truncation, floating point roundoff, Gaussian distributions, etc. We propose *BRECCIA*, a Geospatial Intelligence analysis system, which receives information from humans (as logical sentences), simulations (e.g., weather or environmental predictions), and sensors (e.g. cameras, weather stations, microphones, etc.), where each piece of information has an associated uncertainty; *BRECCIA* then provides responses to user queries based on a new probabilistic logic system which determines a coherent overall response to the query and the probability of that response; this new method avoids the exponential complexity of previous approaches. In addition, *BRECCIA* attempts to identify concrete mechanisms (proposed actions) to acquire new data dynamically in order to reduce the uncertainty of the query response. The basis for this is a novel approach to probabilistic argumentation analysis[1].**

I. Introduction

A. Geospatial Intelligence (GEOINT)

Current knowledge-based GEOINT systems do not incorporate a broad notion of uncertainty quantification (UQ), although such a capability would allow decision makers to make more informed decisions, or to acquire more data before coming to conclusions. In addition, it would be better if system responses were provided with an explanation of how they were derived, as well as how the uncertainty was determined. This can be the result of sensor error, computational error, human error, etc., and the best models should be selected at each time step in order to reduce the variance on quantities of interest. In addition, intelligence, surveillance and reconnaissance support systems should generate dynamic path planning solutions which can include constraints on time, energy, or uncertainty reduction. The automatic generation of constraints arising from the various models can be used to inform the deployment of data measurement systems. The application studied here is UAV (Unmanned Aerial Vehicle) surveillance and reconnaissance in urban areas. Some work has been done in this general area (e.g., see [19] for a novel guidance law in windy urban environments combining pursuit and line-of-sight laws, and [35] for a multi-cost UAV mission path planner).

Exploiting Dynamic Data Driven Application Systems (DDDAS) for large-scale, geographically distributed scenarios promises significant advantages, and we envision an approach that combines various types of information with associated uncertainty to enable model-driven active data acquisition. Figure 1 shows our proposed overall organization of a dynamic data-driven GEOINT application system (called *BRECCIA* after a type of rock formed from several mineral pieces held together in a fine-grained matrix). Typical georeferenced visual and data products include: maps, charts, digital files, imagery and vector information. Value added items include: data verification, correction, updates, densification, reformatting, orthorectification, map finishing, seismic activity, intelligence reports, and additional categories of content [31].

We describe here two major novel research results: (1) the combination of formal probabilistic logic methods with state-of-the-art physics-based uncertainty quantification methods, and (2) uncertainty driven active information data acquisition, demonstrated by UAV path planning, to optimize performance or to resolve contradictory information. The probabilistic logic method is a re-formulation of the approach described in [23] (although Boole [5] first proposed it); see [15] for details. Basically, Nilsson's method requires first solving the SAT problem (i.e., find all consistent truth

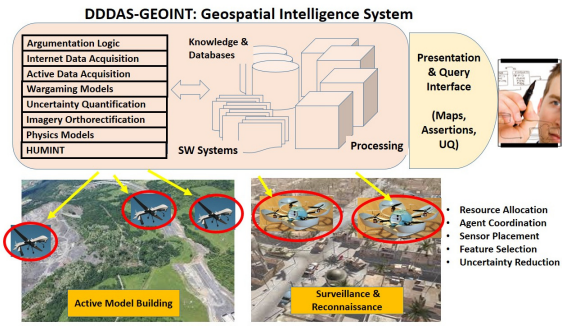

Fig. 1. *BRECCIA*: Uncertainty Quantified GEOINT and Planning.

[1]This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-17-1-0077.

assignments) to set up a set of linear equations, then the use of numerical methods to solve them. We, on the other hand, create a set of nonlinear equations, and solve them directly [15].

BRECCIA is a dynamic uncertainty monitoring and reduction system; that is, uncertainty can arise due to a change in local conditions (e.g., the weather may make movement difficult) or new information may be available (e.g., obscurants in the air, new interesting sites). As a consequence, steps are taken to reduce the uncertainty of assessments; for example, an unmanned aerial vehicle may over fly a zone to get visual confirmation of damage assessments, or a swarm of quadrotors may be sent to provide surveillance or reconnaissance in an urban area. Transparent and precise reasons are offered to the user to explain uncertainty conditions and how they may be resolved.

The study and analysis of geospatial data has progressed from simple Geographic Information Systems (GIS) (spatially organized layers of digital data with associated functional elements [9]) to broader geocomputation systems (see [1], [18]). Such systems involve geospatial data, large-scale computation, and a willingness to apply data mining techniques in order to build models based on experience in addition to analytical theories. This involves GIS as well as artificial intelligence, high performance computing and underlying science models. At its core, this is a major paradigm shift which allows large-scale data exploitation. We believe that GEOINT systems need to be centered in such a modern framework: this means a cloud-based and scalable system with a user-friendly interface (see [22]).

Example applications include decision support for snow removal [34], environmental science [33] knowledge base integration for GIS [24], multiagent systems using GIS data for planning purposes [29], and the exploitation of machine learning and data mining methods in this domain [11], [27], [38], [39] Large-scale geospatial big data projects and systems are relevant as well [7], [13], [32], [37]. We have previously developed the basic computational framework for a DDDAS-based cloud architecture for small-scale structural health monitoring of aircraft [40].

II. The *BRECCIA* System

A. Current Implementation Details

BRECCIA is designed using a well documented multi-agent, Belief-Desire-Intention (BDI) framework called Jason [6]. Jason includes an interpreter for an extended version of the AgentSpeak(L) [28] language, which provides a Prolog-like grammar. Both the style, resembling a natural language application, and the operational semantics of the extended language that enable a data-driven architecture, fit well with the proactive and reactive goals of *BRECCIA*. Each agent in the *BRECCIA* system is composed of a backward chaining inference module (see [25] for a formal justification of modularity in BDI programming languages) with a probabilistic logic component. This module, and probabilistic component, form the most abstract fusion implementation in the *BRECCIA* system. To elaborate, this system may include many functional fusion modules that agents specialize for a particular application. For example, an agent that specializes in managing a particular unmanned air vehicle (UAV) requires a fusion module for estimating target detection. However, the human agent, a "user" in *BRECCIA*, requires information in more abstract terms to support the decision process. For example, a user may query the system by asking for the probability of mission success. Among the many bits of information that support this assertion, with their associated probabilities, and one of which is whether the target was detected, *BRECCIA* would respond as follows:

$$mission\_success[p(0.9), justification([target\_detected, no\_damage, ...])]$$

The probabilistic logic component facilitates the propagation of quantitative uncertainties when making inferences from the knowledge base, all while maintaining the justifications for such inferences. Probabilities of sentences in the knowledge base, which may be interpreted as uncertainty, are represented as beliefs with annotations (a feature of the extended AgentSpeak language). For example, the belief, "Bob thinks it will rain today with 0.9 certainty," may be represented in Jason as follows:

$$will\_rain\_today[p(0.9), source(bob)]$$

If the application requires a data-driven event, for example to notify a user that their base assumptions about a prior simulation have changed, the program simply requires a plan in the following syntax:

$$+will\_rain\_today[p(X)] : X < 0.8 \leftarrow !tell\_bob$$

This plan says that if the certainty for "will rain today" dips below 0.8, then the agent should adopt the goal to tell Bob. This example demonstrates the clarity with which a data-driven application is programmed in this framework. Furthermore, the BDI system allows for the dynamic prioritization of goals (termed intentions once they are adopted by the agent); high priority goals can interrupt low priority intentions and become the current intention. This is a critical feature for a reactive system such as *BRECCIA*; changing assumptions about the current state of the environment should force replanning. In Jason this concept is represented as the plan's "context," shown in the above example after the colon asserting that $X < 0.8$. Another plan may exist in the agent to tell a higher authority when the probability dips below 0.5, for example.

The algorithm for backward chaining with probabilistic logic is shown in Algorithm 1. Inference rules with probabilistic logic are stored as material implications in *BRECCIA* in the following format:

$$mat\_imp(consequent, [antecedents])[p(probability)]$$

As a concrete example, if the implication $a \wedge b \implies c$ should be stored in the knowledge base with probability 0.9, then it is defined in the following syntax:

$$mat\_imp(c, [a, b])[p(0.9)]$$

Data: $L \leftarrow$ List of antecedents
Result: $p(C)$ - The probability of the consequent
$matched\_beliefs \leftarrow [empty\_list]$;
for *each A in L* **do**
  $belief \leftarrow match\_in\_kb(A)$;
  $p(belief) \leftarrow extract\_probability(belief)$;
  $append(matched\_beliefs, belief[p(belief)])$;
end
$p(C) = run\_nlpl(matched\_beliefs)$;
add $C[p(C), justification(mat\_cnf(C)]$ to KB;

Algorithm 1: Backward Chaining with Probabilistic Logic

Algorithm 1 matches each antecedent in the given material implication with a belief in the agent's knowledge base and forms a new list with the current sentence probabilities. If an antecedent is an inferred belief that has yet to be calculated, then the current intention is suspended and a new intention is generated to resolve the constraint. The process of generating intentions for unresolved constraints uses the built-in (non-probabilistic) backward chaining that comes with Jason. However, this process is automatic and simply requires the following plan that responds to unresolved beliefs that have a material implication rule (notice the context after the colon) and adds the goal to run Algorithm 1:

$$+?X : mat\_cnf(X, \_) \leftarrow !backchain(X)$$

Once the list of probabilities have compiled, the non-linear probabilistic logic algorithm (described in section III) is executed to calculate the probability for the inferred belief. Finally the inferred belief is added to the agent's knowledge base. The process of belief-revision (when probabilities of antecedents change) currently utilizes the same algorithm, except first a list of justifications, stored in the belief annotations, is compiled until the most abstract inference is located. Belief-revision in the *BRECCIA* system is an active area of research.

B. Path Planning

BRECCIA includes an RRT* path planner [8], [17] that provides an asymptotically optimal path between two states (in this case between the launch and recovery sites). In this specific implementation, the goal is to fuse uncertainties in the environment and vehicle models into an estimate of the probability of success of a selected path in terms of power, flight time constraints, etc. Consider a simplified case of the Raven UAV tasked with monitoring a location in an urban environment.

The *simulate_path* method calculates the cost of the path between two locations in an environment at incremental steps during the path planning procedure. In the current implementation the cost is a representation of the flight time required to traverse between the two locations in one-meter increments. In this simplified model, the component of drag in the direction of the travel is added to the vehicle's velocity to calculate a ground speed. The path cost is then calculated from the ground speed, hence higher ground speeds are favorable.

Data: $v_1 \leftarrow start\_position; v_2 \leftarrow end\_position$
Result: $path\_cost$
$path\_velocity \leftarrow v_2 - v_1$;
$air\_velocity \leftarrow cruise\_speed * \frac{path\_velocity}{|path\_velocity|}$;
for *each meter in path* **do**
  $wind\_velocity \leftarrow sample\_wind(position)$;
  $ground\_speed \leftarrow$
   $cruise\_speed + \frac{wind\_velocity \cdot air\_velocity}{|air\_velocity|}$;
  $time\_burned \leftarrow (ground\_speed)^{-1}$;
  if $\frac{time\_burned}{60} > max\_flight\_time$ **then**
    $path\_cost \leftarrow inf$;
    $break$;
  else
    $path\_cost \leftarrow path\_cost + time\_burned$;
  end
end

Algorithm 2: Raven Path Cost Algorithm.

The critical step in Algorithm 2 is sampling from the wind model. Data from the wind model is captured in Matlab's *griddedInterpolant* object to enable the path planner to sample from any location. Each wind sample is purposely corrupted by Gaussian noise with the given variance from the wind model as determined in the vortex simulation (see below). *BRECCIA* then runs the path planner multiple times to calculate a variance in path costs. Figure 2 shows the final path discovered by RRT using vortex simulation data.

The path planner was run thirty times for the operational scenario shown in Figure 2 with an assumed wind model variance of 0.5. One example of a resulting path is shown in Figure 2 as a dashed line between waypoints. A histogram of resulting flight times and a fitted normal distribution is shown in Figure Figure 3. Based on the resulting model, the mean flight time is approximately 612 seconds with a standard deviation of 27 seconds. The resulting 90^{th} percentile flight-time is 653 seconds and is shown marked on the cumulative distribution function in Figure 4. These values are propagated to *BRECCIA*'s argumentation system for uncertainty fusion.

The key attribute of this path-sampling strategy is that the uncertainty in path optimality is included in the resulting flight-time model. Hence, the parameters that control RRT*, such as the number of iterations, may be adjusted by *BRECCIA* to achieve more or less uncertainty in the final result. The benefit of this is the ability to re-plan in real-time while maintaining awareness of the probability of mission success.

C. Vortex (Cavity Flow) Modeling

As part of the mission planning, the weather must be considered, and Vortex Modeling provides initial wind velocity estimates and their probabilities. These wind velocities are

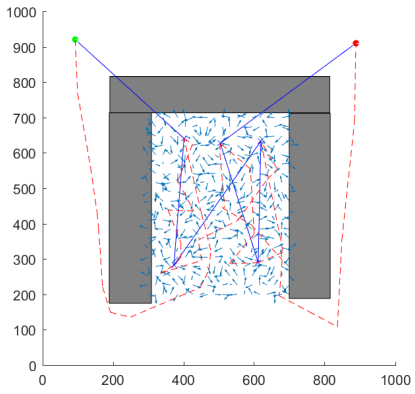

Fig. 2. Operational Scenario and Final Path Calculated by RRT*.

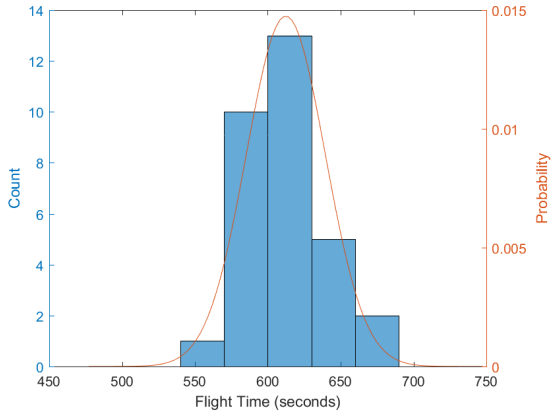

Fig. 3. Histogram of Path Flight Times and Fitted Normal Distribution.

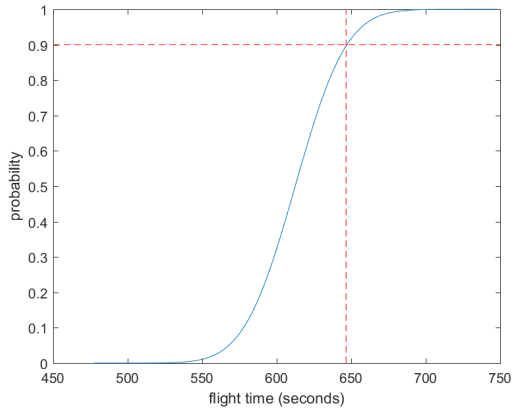

Fig. 4. Cumulative Distribution of Flight Times with 90th Percentile Marked.

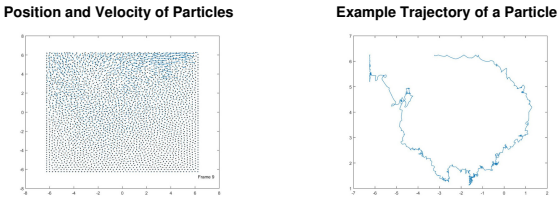

Fig. 5. The State of Particles after 12000 Steps (left); a Sample Particle Trajectory (right).

initially provided to the path planning algorithm by means of a simple 2D particle model simulation. The approach is based on the detailed description given by Greenspan [14]. It is assumed that the air mass is comprised of N particles, $\vec{P}_i$, $i = 1 \ldots N$, each with mass m. A system of coupled ODEs describes the motion of each particle:

$$\vec{F}_i = m\frac{\partial^2 r}{\partial t^2}, i = 1 \ldots N$$

where $\vec{F}_i$ is the force on particle i, and $\vec{r}_i$ is the position vector of particle i. Note that:

$$\vec{F}_i = \vec{F}_i^{**} + \vec{F}_i^{*}$$

where $\vec{F}_i^{**}$ is a long range force (gravity and $g = 980$), and $\vec{F}_i^{**}$ is a short range force that holds within specified distance D:

$$\vec{F}^*_{ij,k} = [-\frac{G}{(r_{ij,k})^p} + \frac{H}{(r_{ij,k})^q}]\frac{\vec{r}_{ij,k}}{r_{ij,k}}$$

To obtain values for the mission simulation, 2576 points are used in a square area where it is assumed that three sides are closed and one open (the top). A wind ($V = [-10, 0]$) passing by the top produces the cavity flow. The parameters are set to: $G = 0$, $p = 3$, $H = 100$, $q = 5$, $D = 0.35$ (the initial distance between particles is 0.25), and $\delta t = 0.0001$. A snapshot of the state of the particles and the trajectory of a sample particle after 12,000 steps in the simulation are shown in Figure 5 on the left and right, respectively. A Gaussian noise model on the individual particle forces is used with $\sigma^2 = 0.0001$.

III. Probabilistic Logic

Here we address the problem of finding a suitable representation for uncertainty associated with logical sentences. Although several approaches have been proposed in the past (see [2], [10], [12], [16], [21], [26], [36], [30]), they generally have some significant drawbacks. Usually, these have to do with the computational complexity of the semantics of the sentences (i.e., finding the set of consistent truth assignments is exponential in the number of sentences, or for Domingos, exponential in the number of cliques in the Markov graph [3]).

We have developed a new approach which computes the probabilities of the atoms in the sentences, and in terms of these, provides a solution for $Pr(Q \mid KB)$, where Q is the query and KB is the knowledge base set of sentences (see [15] for details). Moreover, the knowledge of the probabilities of the atoms allows us to determine where the most uncertain part of the argument lies, and to

allocate resources to lower that uncertainty, thus decreasing the uncertainty of the query. This is done by exploiting the probability of a disjunctive clause, and developing a set of equations from the sentences and their probabilities, and then solving those equations (the number of equations equals the number of sentences).

Our approach to probabilistic logic starts with an analysis of Nilsson's method [23][2]. Given a set of n sentences, $S = \{S_1, S_2, \ldots, S_n\}$, in the propositional calculus, where $\{S_1, \ldots, S_{n-1}\}$ is the KB and S_n is the query, he first finds the set of models of the sentences (i.e., the set of truth value assignments to the sentences that are consistent using the general semantic tree [20] for a set of sentences). In our new approach [15], we avoid the exponential complexity of most other algorithms by solving for the logical variable (atom) probabilities directly as follows.

First, we assume that the sentences are given in conjunctive normal form. This means that each sentence is a disjunct of literals (an atom or its negation). Our second assumption is that $Pr(P \wedge Q) = Pr(P)Pr(Q)$; note that if this assumption is violated, our methods also allow the bounds on the probability to be determined. Next, we find the set of logical atoms (i.e., variables) in S; let $A = \{A_1, A_2, \ldots, A_k\}$ be this set. In this case the probability of a sentence can be computed from the probability of its literals as follows:

$$Pr(L_1 \vee L_2 \vee \ldots \vee L_p) =$$
$$Pr(L_1) + Pr(L_2 \vee \ldots \vee L_p)$$
$$-Pr(L_1)Pr(L_2 \vee \ldots \vee L_p),$$

where the probabilities of clauses on the right hand side are computed recursively.

Assuming that the logical (random) variables are independent, each sentence gives rise to a (usually) nonlinear equation defined by the recursive probability of the disjunctive clause as defined above. The resulting set of equations can be solved using standard nonlinear solvers (e.g., *fsolve* in Matlab), and a set of consistent values for the probabilities of the atoms determined. Of course, one problem with the nonlinear solver approach is that it may not find a solution, even when one or more exist. Thus, our current approach is to solve all equations that have a single unknown (recursively), and then use an iterative method to find a set of atom probabilities which produce the correct sentence probabilities.

IV. Experiments

Here we describe a scenario which uses a Raven, man-portable, hand-launched small unmanned aerial vehicle (see Figure 6). The mission is described in Figure 7 and consists of going to a specified location (Named Area of Interest - NAI), loitering there while reconnoitering some points of interest, then going to the recovery location. Note that the area is similar to our simulation scenario in that there are three closed sides formed by the buildings, and the fourth side is open. Assume the KB has sentences related to the use of several Raven platforms for a mission, and a subset of KB sentences are extracted that form an argument for using the specific platform called Raven_1; then the argument sentences and their origins are as follows (the sources of information, i.e., which *BRECCIA* component produced them are given in parentheses):

Fig. 6. The Raven UAV.

1. Raven_1 Platform Available (Maintenance Reports)
2. Raven_1 Air Control Measures OK (Mission Plan)
3. Wind<17 Knots (Weather Report)
4. Precipitation Low (Weather Report)
5. Visibility OK (Weather Report)
6. Temperature between [0,90] (Weather Report)
7. (3 $\wedge$ 4 $\wedge$ 5 $\wedge$ 6) $\rightarrow$ Weather_OK
8. Target-Loiter distance $<$ 7miles (Mission Plan)
9. Raven_1 Electro-Optical (Mission Plan)
10. Raven_1 Infra-Red (Mission Plan)
11. (9 and 10) $\rightarrow$ Collection Requirement Done
12. Raven_1 Power OK (Path Planning)
13. Raven_1 Battery OK (Maintenance Reports)
14. Raven_1 Speed Known (Path Planning)
15. Raven_1 Altitude Known (Path Planning)
16. Raven_1 Loiter Time Known (Path Planning)
17. (12 $\wedge$ 13 $\wedge$ 14 $\wedge$ 15 $\wedge$ 19 $\wedge$ 16 $\wedge$ 21) $\rightarrow$ Path OK
18. Raven_1 Crew Available (Mission Plan)
19. Raven_1 Route Time Known (Path Planning)
20. Air Defense Threat Known (Mission Plan)
21. Named Areas of Interest Defined (Mission Plan)
22. (1 $\wedge$ 2 $\wedge$ Weather_OK $\wedge$ Collection Requirements Specified $\wedge$ Path_OK $\wedge$ 18 $\wedge$ 20) $\rightarrow$ Raven_1_Mission_OK
23. (Query) Raven_1_Mission_OK?

Note that sentences 7, 11, 17, and 22 are human specified rules. This leads to 23 CNF clauses for sentences 1 to 22, and 1 for the query. Note that the probabilities for the individual sentences come from either human attribution (e.g., Raven_1 platform available), or from noise models in the data (velocity vectors have Gaussian noise as determined by the simulations). Furthermore, note that some sentences are comprised of only a single literal, and thus, the probability of the associated atom is the probability of the sentence. However, the probability of some atoms (e.g., Path OK in sentence 17) is implicit and must be found as part of the

[2]Note that Nilsson's method for propositional calculus is the same as that proposed by George Boole in the 1800's [4], [5].

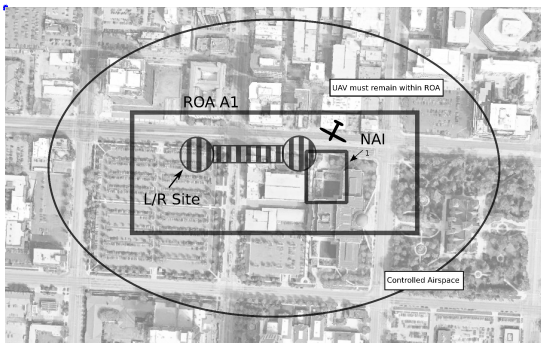

Fig. 7. An Example Mission.

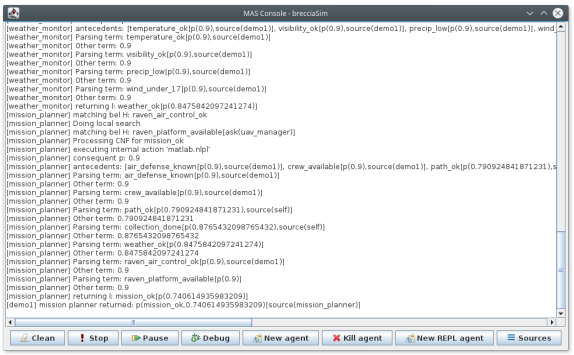

Fig. 8. *BRECCIA* Mission Simulation Output

solution for the set of nonlinear equations arising from the sentences. The probabilities for the path planning sentences arise from simulations as shown in Figure 2.

As a preliminary test of the probabilistic logic computation, all KB sentences were assigned the (same) value ranging from 0.9 to 1.0 in steps of 0.02, and the resulting probabilities assigned to the query were [0.7406 0.8373 0.9011 0.9452 0.9768 1.0000]. As can be seen, when all the sentences are true (probability 1), the query is true with probability 1.

Now consider the case where sentences are provided by the following specialized agents in a *BRECCIA* system and their assigned sentences.

$$(mission\_planner,$$
$$uav\_manager,$$
$$weather\_monitor) \in Breccia$$
$$(2, 8, 9, 10, 11, 12,$$
$$13, 15, 16, 17, 18, 19, 22) \in mission\_planner$$
$$(1, 13) \in uav\_manager$$
$$(3, 4, 5, 6, 7) \in weather\_monitor$$

To facilitate inference across agents, material implications that include external propositions include the special annotation "ask(agent)". Intentions are then automatically generated to query agents across the *BRECCIA* network. A mission is simulated with every sentence probability set to 0.9, and Figure 8 shows the simulation console output. *BRECCIA* finds the following values for the implicit atoms: Pr(Weather OK) = 0.8476, Pr(Collection Requirement Done) = 0.8765, Pr(Path OK) = 0.7909, and Pr(Mission OK) = 0.7406. Thus, to increase the probability of mission success, it is essential to increase the probability of Path OK. For example, replanning to ensure that the weather info in sentences 3–6 and power info in 12–13 has probability 0.95, raises the Pr(Mission OK) to 0.7560. The user must decide if it is worth investing resources to improve that information. The advantage of *BRECCIA*, is that deeper insight into the reasons for the overall probability of success can be known.

V. Conclusions and Future Work

BRECCIA, a dynamic geospatial information analysis system, is described which provides a unified probabilistic framework for multi-source data uncertainty. The major contributions here are: (1) an effective probabilistic logic methodology, and (2) an experimental GEOINT system which allows the specification and combination of uncertain data from a wide variety of information sources, including the ability to determine specific actions to lower the uncertainty of the likelihood of statements of interest.

Future work includes:

- the extension of the knowledge base to first order logic,
- a more in-depth demonstration of the argumentation capabilities of *BRECCIA*,
- the addition of other information services (e.g., the use of available 3D urban wind models),
- the inclusion of cost models to provide cost-benefit analysis for the user in making decisions.
- more advanced belief revision.
- the further improvement of the interaction between *BRECCIA* and the information sources, and
- field testing with UAV reconnaissance missions.

References

[1] R.J. Abraham and L. See, editors. *GeoComputation*. CRC Press, Boca Raton, FL, 2014.

[2] T. Alsinet, C.I. Chesnevar, L. Godo, and G.R. Simari. A Logic Programming Framework for Possibilistic Argumentation. *Fuzzy Sets and Systems*, 159(10):1208–1228, 2008.

[3] M. Biba. *Integrating Logic and Probability: Algorithmic Improvements in Markov Logic Networks*. PhD thesis, University of Bari, Bari, Italy, 2009.

[4] G. Boole. Further Observations on the Theory of Probabilities. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2:96–101, 1851.

[5] G. Boole. *An Investigation of the Laws of Thought*. Walton and Maberly, London, UK, 1857.

[6] R.H. Bordini, J.F. Huebner, and M. Wooldridge. *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley Series in Agent Technology. Wiley, Hoboken, NJ, 2007.

[7] S. Bowers, E. Jaeger-Frank, B. Broderic, and C. Baru. Managing Scientific Data: from Data Integration to Scientific Workflows. In A.K. Sinha, editor, *Geoinformatics: Data to Knowledge*, pages 109–130, Boulder, CO, 2006. The Geological society of America.

[8] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, and S. Thrun. *Principles of Robot Motion*. MIT Press, Cambridge, MA, 2005.

[9] J. Delaney and K. van Niel. *Geographical Information systems*. Oxford University Press, Oxford, UK, 2007.

[10] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan and Claypool, San Rafael, CA, 2009.

[11] N. Gilardi and S. Bengio. Comparison of Four Machine Learning Algorithms for Spatial Data Analysis. In G. Dubois, J. Malczewski, and M. De Cort, editors, *Mapping Radioactivity in the Environment – Spatial Interpolation Comparison*, volume 97, pages 222–237, Luxembourg, May 2003. Office for Official Publications of the European Communities.

[12] V. Gogate and P. Domingos. Probabilistic Theorem Proving. *Communications of the ACM*, 59(7):107–115, 2016.

[13] D.G. Goulinas. Development of a GIS-based System for Highway Analysis. In C.A. Brebbia and P. Pascolo, editors, *GIS Technologies and Their Environmental Applications*, pages 167–176, Southhampton, UK, 1998. Computational Mechanics Pubs.

[14] D. Greenspan. *Particle Models*. Birkhäuser, Boston, MA, 1997.

[15] T.C. Henderson, A. Mitiche, R. Simmons, and X. Fan. A Preliminary Study of Probabilistic Argumentation. Technical Report UUCS-17-001, University of Utah, February 2017.

[16] A. Hunter. A Probabilistic Approach to Modelling Uncertain Logical Arguments. *International Journal of Approximate Reasoning*, 54(1):47–81, January 2013.

[17] Sertac Karaman and Emilio Frazzoli. Incremental Sampling-based Algorithms for Optimal Motion Planning. In Y. Matsuoka, H. Durrant-Whyte, and J. Neira, editors, *Proceedings of Robot Science and Systems*, Zaragoza, Spain, 2010.

[18] H.A. Karimi, editor. *Big Data: Techniques and Technologies in Geoinformatics*. CRC Press, Boca Raton, FL, 2014.

[19] M. Kothari, I. Postlewaite, and D.-W. Gu. UAV Path Following in Windy Urban Environments. *Journal of Intelligent and Robotic Systems*, 74:1013–1028, 2014.

[20] R. Kowalsi and P.J. Hayes. Semantic Trees in Automatic Theorem Proving. In J.J. Siekmann and G. Wrightson, editors, *Automation of Reasoning*, pages 217–232, Berlin, 1983.

[21] H. Li, N. Oren, and T. Norman. Probabilistic Argumentation Frameworks. In *Proc. 1st International Workshop on the Theory and Applications of Formal Argumentation*, Beijing, China, August 2011.

[22] B. Liu, Y. Chen, A. Hadiks, E. Blasch, A. Aved, D. Shen, and G. Chen. Information Fusion in a Cloud Computing Era: A Systems-Level Perspective. *IEEE Aerospace and Electronic Systems Magazine*, 29(10):16–24, October 2014.

[23] N. Nilsson. Probabilistic Logic. *Artificial Intelligence Journal*, 28:71–87, 1986.

[24] D. Nute, W.D. Potter, Z. Cheng, M. Dass, A. Glende, F. Maierv, C. Routh, H. Uchiyama, J. Wang, S. Witzig, M. Twery, P. Knopp, S. Thomasma, and H.M. Rauscher. A Method for Integrating Multiple Components in a Decision Support System. *Computers and Electronics in Agriculture*, 49:44–59, 2005.

[25] G. Ortiz-Hernández, J.F. Hübner, R.H. Bordini, A. Guerra-Hernández, G.J. Hoyos-Rivera, and N. Cruz-Ramírez. A Namespace Approach for Modularity in BDI Programming Languages. In M. Baldoni, J.P. Müller, I. Nunes, and R. Zalila-Wenkstern, editors, *Engineering Multi-Agent Systems: 4th International Workshop, EMAS 2016, Singapore, Singapore, May 9-10, 2016, Revised, Selected, and Invited Papers*, pages 117–135, Cham, 2016. Springer International Publishing.

[26] L. De Raedt, A. Kimmig, and H. Toivonen. HProbLog: A Probabilistic Prolog and its Application in Link Discovery. pages 2462–2467, 2007.

[27] R. Ramachandran, J. Rushing, X. Li, C. Kamath, H. Conover, and S. Graves. Bird's-eye View of Data Mining in Geosciences. In A.K. Sinha, editor, *Geoinformatics: Data to Knowledge*, pages 235–248, Boulder, CO, 2006. The Geological society of America.

[28] A.S. Rao. AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. In W. Van de Velde and J.W. Perram, editors, *Agents Breaking Away: 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW '96 Eindhoven, The Netherlands, January 22–25, 1996 Proceedings*, pages 42–55, Berlin, Germany, 1996. Springer Verlag.

[29] F.I. Riadh, S.E. Karim, B. Solaiman, and B.A. Mohamed. Analyzing Spatial-Temporal Geographic Information based on Blackboard Architecture and Multi-Agent system. *International Journal of Computer Science and Network Security*, 6(8A):4–10, August 2006.

[30] M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62(1-2):107–136, February 2006.

[31] C.M. Scaparrotti. *Geospatial Intelligence in Joint Operations*. Number Joint Publication 2-03. US Government, Washington, DC, 2012.

[32] M.H. Sharker and H.A. Karimi. Distributed and Parallel Computing. In H.A. Karimi, editor, *Big Data: Techniques and Technologies in GeoInformatics*, pages 2–31, Boca Raton, FL, 2014. CRC Press.

[33] J. Song, B. Xiang, X. Wang, L. Wu, and C. Chang. Application of Dynamic Data Driven Application System in Environmental Science. *Environmental Review*, 22:287–297, 2014.

[34] R. Sugumaran, S. Ilavajhala, and V. Sugumaran. Development of a Web-Based Intelligent Spatial Decision Support System (WEBISDSS): A Case Study with Snow Removal Operations. In B.N. Hilton, editor, *Emerging Spatial Information Systems and Applications*, pages 184–202, London, 2007. Idea Group Publishing.

[35] L.D. Swartzentrubre. *Improving Path Planning of Unmanned Aerial Vehicles in an Immersive Environment using MetaPaths and Terrain Information*. PhD thesis, Iowa State University, 2009.

[36] M. Thimm. A Probabilistic Semantics for Abstract Argumentation. In *Proc. 20th European Conference on Artificial Intelligence*, Montpellier, France, August 2012.

[37] M.-H. Tsou. Bridging the Gap: Connecting Internet-Based Spatial Decision Support Systems to the Field-Based Personnel with Real Time Wireless Mobile GIS Applications. In S. Balram and S. Dragicevic, editors, *Collaborative Geographic Information Systems*, pages 286–315, Hershey, PA, 2006. Idea Group Publishing.

[38] T. van Zyl. Machine Learning on Geospatial Big Data. In H.A. Karimi, editor, *Big Data: Techniques and Technolgies in GeoInformatics*, pages 134–148, Boca Raton, FL, 2014. CRC Press.

[39] R.R. Vatsavai, A. Ganguly, V. Chandola, A. Stefanidis, S. Klasky, and S. Shekhar. Spatiotemporal Data Mining in the Era of Big Data Spatial Data: Algorithms and Applications. In *Proc. 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data*, pages 1–10, NY, NY, 2012. ACM.

[40] W. Wang, A. Joshi, N. Tirpankar, P. Erickson, M. Cline, P. Thangaraj, and T.C. Henderson. Bayesian Computational Sensor Networks: Small-scale Structural Health Monitoring. In *Proceedings of the International Conference on Computational Science*, ICCS '15, Reykavik, Iceland, June 2015. Springer-Verlag.