INTERNATIONAL JOURNAL OF RESEARCH IN MECHANICAL, MECHATRONICS AND AUTOMOBILE ENGINEERING (IJRMMAE)
ISSN: 2454-1435 (Print) | 2454-1443 (online)
Volume 4 Issue 1 – www.ijrmmae.in – Pages 29-33

MUTLISENSOR DATA FUSION METHODS AND ARCHITECTURE

S. Chandravadhana[1], R. Nandhakumar[2]

Department of Mechatronics Engineering, Agni College of Technology, Chennai.

ABSTRACT

Multisensor data fusion is the process of combining observations from a number of different sensors to provide a robust and complete description of an environment or process of interest. Data fusion finds wide application in many areas of robotics such as object recognition, environment mapping, and localization. This paper explains the various methods and architectures. Most current data fusion methods employ probabilistic descriptions of observations and processes and use Bayes' rule to combine this information. This chapter surveys the main probabilistic modeling and fusion techniques including grid-based models, Kalman filtering, and sequential Monte Carlo techniques. This chapter also briefly reviews a number of non-probabilistic data fusion methods. Data fusion systems are often complex combinations of sensor devices, processing, and fusion algorithms. This chapter provides an overview of key principles in data fusion architectures from both a hardware and algorithmic viewpoint. The applications of data fusion are pervasive in robotics and underly the core problem of sensing, estimation, and perception. The essential algorithmic tools of data fusion are reasonably well established.

MULTISENSOR DATA FUSION METHODS

The most widely used data fusion methods employed in robotics originate in the fields of statistics, estimation, and control. However, the application of these methods in robotics has a number of unique features and challenges. In particular, autonomy is most often the goal and so results must be presented and interpreted in a form from which autonomous decisions can be made, for recognition or navigation, for example. In this section we review the main data fusion methods employed in robotics. These are very often based on probabilistic methods, which are indeed now considered the standard approach to data fusion in all robotics applications. Probabilistic data fusion methods are generally based on Bayes' rule for combining prior and observation information. Practically, this may be implemented in a number of ways: through the use of the Kalman and extended Kalman filters, through sequential Monte Carlo methods, or through the use of functional density estimates. Each of these is reviewed. There are a number of alternatives to probabilistic methods. These include the theory of evidence and interval methods. Such alternative techniques are not as widely used as they once were, however they have some special features that can be advantageous in specific problems. These, too, are briefly reviewed.

BAYESIAN FILTERING

Filtering is concerned with the sequential process of maintaining a probabilistic model for a state which evolves over time and which is periodically observed by a sensor. Filtering forms the basis for many problems in tracking and navigation. The general filtering problem can be formulated in Bayesian form. This is significant because it provides a common representation for a range of discrete and continuous data fusion problems without recourse to specific target or observation models. Define **x**t as the value of a state of interest at time t. This may, for example, describe a feature to be tracked, the state of a process being monitored, or the location of a platform for which navigation data is required. For convenience, and without loss of generality, time is defined at discrete (asynchronous) times tk _ k.

INTERNATIONAL JOURNAL OF RESEARCH IN MECHANICAL, MECHATRONICS AND AUTOMOBILE ENGINEERING (IJRMMAE)
ISSN: 2454-1435 (Print) | 2454-1443 (online)
Volume 4 Issue 1 – www.ijrmmae.in – Pages 29-33

PROBABILISTIC GRIDS

Probabilistic grids are conceptually the simplest way of implementing Bayesian data fusion methods. They can be applied both to problems in mapping [3, 4] and tracking [5]. In mapping applications, the environment of interest is divided into a grid of equally sized spatial cells. Each cell is indexed and labeled with a property, thus the state **x**ij may describe a two-dimensional world indexed by ij and having the property x. Interest is focused on maintaining a probability distribution on possible state values P(**x**ij) at each grid cell. Typically, in navigation and mapping problems, the property of interest has only two values O and E, occupied and empty, respectively, and it is then usual to assume that P(**x**ij = O) = 1− P(**x**ij = E). However, there is no particular constraint on the property encoded by the state **x**ij which could have many values (green, red, blue, for example) and indeed be continuous (the temperature at a cell, for example). Once the state has been defined, Bayesian methods require that a sensormodel or likelihood function for the sensor be established. In theory, this requires specification of a probability distribution P(**z** | **x**ij = xij)mapping each possible grid state xij to a distribution on observations. Practically, however, this is implemented simply as another observation grid so that for a specific observation **z** = z (taken from a specific location), a grid of likelihoods on the states **x**ij is produced in the form P(**z** = z | **x**ij) = Λ(**x**ij).

It is then trivial to apply Bayes' rule to update the property value at each grid cell as P+(**x**ij) = CΛ(**x**ij)P(**x**ij) , ∀i, j , (25.9) whereC us a normalizing constant obtained by summing posterior probabilities to one at node ij only. Computationally, this is a simple pointwise multiplication of two grids. Some care needs to be taken that the two grids appropriately overlap and align with each other at the right scale. In some instances it is also valuable to encode the fact that spatially adjacent cells will influence each other; that is, if we knew the value of the property (occupancy, temperature, for example) at ij we will have some belief also of the value of this property at adjacent nodes i +1, j, i, j +1, etc. Different sensors and the fusion of different sensor outputs is accommodated simply by building appropriate sensor models Λ(**x**ij). Grids can also be used for tracking and self-tracking (localization). The state **x**ij in this case is the location of the entity being tracked. This is a qualitatively different definition of state from that used in mapping. The probability P(**x**ij) must now be interpreted as the probability that the object being tracked occupies the grid cell ij. In the case of mapping, the sum of property probabilities at each grid cell is one, whereas in the case of tracking, the sum of location probabilities over the whole grid must sum to one. Otherwise, the procedure for updating is very similar.

An observation grid is constructed which, when instantiatedwith an observation value, provides a location likelihood grid P(**z** = z | **x**ij) = Λ(**x**ij). Bayes' rule is then applied to update the location probability at each grid cell in the same form as except that now the normalization constant C is obtained by summing posterior robabilities over all ij grid cells. This can become computationally expensive, especially if the grid has three or more dimensions. One major advantage of grid-based tracking is that it is easy to incorporate quite complex prior information. For example, if it is known that the object being tracked is on a road, then the probability location values for all off-road grid cells can simply be set to zero. Grid-based fusion is appropriate to situations where the domain size and dimension are modest. In such cases, grid-based methods provide straightforward and effective fusion algorithms. Grid-based methods can be extended in a number of ways; to hierarchical (quadtree) grids, or to irregular (triangular, pentagonal) grids. These can help reduce computation in larger spaces. Monte Carlo and particle filtering methods may be considered as grid-based methods, where the grid cells themselves are sample of the underlying probability density for the state.

KALMAN FILTER

The Kalman filter is a recursive linear estimator that successively calculates an estimate for a continuous valued state, that evolves over time, on the basis of periodic observations of the state. The

Kalman filter employs an explicit statistical model of how the parameter of interest $\mathbf{x}(t)$ evolves over time and an explicit statistical model of how the observations $\mathbf{z}(t)$ that are made are related to this parameter. The gains employed in a Kalman filter are chosen to ensure that, with certain assumptions about the observation and process models used, the resulting estimate $\hat{\mathbf{x}}(t)$ minimizes mean-squared error and is thus the conditional mean $\hat{\mathbf{x}}(t) = E[\mathbf{x}(t) \mid \mathbf{Z}^t]$: an average, rather than a most likely value. The Kalman filter has a number of features which make it ideally suited to dealing with complex multisensory estimation and data fusion problems. In particular, the explicit description of process and observations allows a wide variety of different sensor models to be incorporated within the basic algorithm. In addition, the consistent use of statistical measures of uncertainty makes it possible to quantitatively evaluate the role each sensor plays in overall system performance. Further, the linear recursive nature of the algorithm ensures that its application is simple and efficient. For these reasons, the Kalman filter has found widespread application in many different data fusion problems [6–9]. In robotics, the Kalman filter is most suited to problems in tracking, localization, and navigation, and less so to problems in mapping. This is because the algorithm works best with well-defined state descriptions (positions, velocities, for example), and for states where observation and time-propagation models are also well understood.

MULTISENSOR FUSION ARCHITECTURES

The multisensor fusion methods described in the previous section provide the algorithmic means by which sensor data and their associated uncertainty models can be used to construct either implicit or explicit models of the environment. However, a multisensor fusion system must include many other functional components to manage and control the fusion process. The organization of these is termed a multisensor fusion architecture.

Architectural Taxonomy

Multisensor systems architectures can be organized in various ways. The military community has developed a layout of functional architectures based on the joint directors of the laboratories (JDL) model for multisensory systems. This approach views multisensor fusion in terms of signal, feature, threat and situation analysis levels (so-called JDL levels). The assessment of such systems is specified in terms of tracking performance, survivability, efficiency and bandwidth. Such measures are not generally appropriate in robotics applications and so the JDL model is not discuss this further here (see [17,18] for details).Other classification schemes distinguish between low- and high-level fusion [19], or centralized versus decentralized processing or data versus variable [20]. A general architectural framework for multisensory robotic systems has been developed and described in detail by Makarenko [21], and we will base our discussion on this approach. A system architecture is defined as follows.

Meta-architecture. A set of high-level considerations that strongly characterize the system structure. The selection and organization of the system elements may be guided by aesthetics, efficiency, or other design cri- teria and goals (for example, system and component comprehensibility, modularity, scalability, portability, interoperability, (de)centralization, robustness, fault tolerance).

Algorithmic Architecture. A specific set of information fusion and decision-making methods. These methods address data heterogeneity, registration, calibration, consistency, information content, independence, time interval and scale, and relationships between models and uncertainty.

Conceptual Architecture. The granularity and functional roles of components (specifically, mappings from algorithmic elements to functional structures).

Logical Architecture. Detailed canonical component types (i. e., object-oriented specifications) and interfaces to formalize intercomponent services. Components may be ad hoc or regimented, and other

INTERNATIONAL JOURNAL OF RESEARCH IN MECHANICAL, MECHATRONICS AND AUTOMOBILE ENGINEERING (IJRMMAE)
ISSN: 2454-1435 (Print) | 2454-1443 (online)
Volume 4 Issue 1 – www.ijrmmae.in – Pages 29-33

concerns include granularity, modularity, reuse, verification, data structures, semantics, etc. Communication issues include hierarchical versus heterarchical organization, shared memory versus message passing, information based characterizations of subcomponent interactions, pull/push mechanisms, subscribe–publish mechanisms, etc. Control involves both the control of actuation systems within the multisensor fusion system, as well as control of information requests and dissemination within the system, and any external control decisions and commands.

Execution Architecture. Defines mapping of components to execution elements. This includes internal or external methods of ensuring correctness of the code (i. e., that the environment and sensor models have been correctly transformed from mathematical or other formal descriptions into computer implementations), and also validation of the models (i. e., ensure that the formal descriptions match physical reality to the required extent).

In any closed-loop control system, sensors are used to provide the feedback information describing the current status of the system and its uncertainties. Building a sensor system for a given application is a system engineering process that includes the analysis of system requirements, a model of the environment, the determination of system behavior under different conditions, and the selection of suitable sensors. The next step in building the sensor system is to assemble the hardware components and develop the necessary software modules for data fusion and interpretation. Finally, the system is tested, and the performance is analyzed. Once the system is built, it is necessary to monitor the different components of the system for the purpose of testing, debugging, and analysis.

APPLICATIONS

Multisensor fusion systems have been applied to a wide variety of problems in robotics (see the references for this chapter), but the twomost general areas are dynamic system control and environment modeling. Although there is some overlap between these, they may generally be characterized as

- **Dynamic system control:** the problem is to use appropriate models and sensors to control the state of a dynamic system (e.g., industrial robot, mobile robot, autonomous vehicle, surgical robot). Usually such systems involve real-time feedback control loops for steering, acceleration, and behavior selection. In addition to state estimation, uncertainty models are required. Sensors may include force/torque sensors, gyros, global positioning system (GPS), position encoders, cameras, range finders, etc.;
- **Environment modeling:** the problem is to use appropriate sensors to construct a model of some aspect of the physical environment. This may be a particular object, e.g., a cup, a physical part, a face, or a larger part of the surroundings, e.g., the interior of a building, part of a city or an extended remote or underground area. Typical sensors include cameras, radar, 3-D range finders, infrared (IR), tactile sensors and touch probes (CMMs), etc. The result is usually expressed as geometry (points, lines, surfaces), features (holes, sinks, corners, etc.), or physical properties. Part of the problem includes the determination of optimal sensor placement.

CONCLUSION

Multisensor data fusion has progressed greatly in the last few decades; further advances in the field will be documented in the robotics and multisensor fusion and integration conference and journal literature. Robust applications are being fielded based on the body of theory and experimental knowledge produced by the research community. Current directions of interest include:

1. Large-scale, ubiquitous sensor systems
2. Bio-based or biomimetic systems

ISSN: 2454-1435 (Print) | 2454-1443 (online)
Volume 4 Issue 1 – www.ijrmmae.in – Pages 29-33

3. Medical in situ applications
4. Wireless sensor networks

Representative large-scale examples include intelligent vehicle and road systems, as well as instrumented contexts such as cities. Biological principles may provide fundamentally distinct approaches to the exploitation of dense, redundant, correlated, noisy sensors, especially when considered as part of a Gibbsian framework for behavioral response to environmental stimuli. Another issue here is the development of a theoretical understanding of sensor system development, adaptivity, and learning with respect to the particular context in which the system is deployed.

REFERENCES

[1] S. Thrun, W. Burgard, D. Fox: Probabilistic Robotics (MIT Press, Cambridge 2005)
[2] J.O. Berger: Statistical Decision Theory and Bayesian Analysis (Springer, Berlin, Heidelberg 1985)
[3] A. Elfes: Sonar-based real-world mapping and navigation, IEEE Trans. Robot. Autom. **3**(3), 249–265 (1987)
[4] A. Elfes: Integration of sonar and stereo range data using a grid-based representation, Proc. IEEE Int. Conf. Robot. Autom. (1988) pp. 727– 25.5 L.D. Stone, C.A. Barlow, T.L. Corwin: Bayesian Multiple Target Tracking (Artech House, Norwood 1999)
[5] Y. Bar-Shalom: Multi-Target Multi-Sensor Tracking (Artec House, Norwood 1990)
[6] Y. Bar-Shalom, T.E. Fortmann: Tracking and Data Association (Academic, New York 1988)
[7] P.S. Maybeck: Stochastic Models, Estimaton and Control, Vol. I (Academic, New York 1979)
[8] W. Sorensen: Special issue on the applications of the Kalman filter, IEEE Trans. Autom. Control **28**(3),254–255 (1983)
[9] B.D.O. Anderson, J.B. Moore: Optimal Filtering (Prentice Hall, Upper Saddle River 1979)
[10] J. Manyika, H.F. Durrant-Whyte: Data Fusion and Sensor Management: An Information-Theoretic Approach (Prentice Hall, Upper Saddle River1994)
[11] S. Sukkarieh, E. Nettleton, J.H. Kim, M. Ridley, A. Goktogan, H. Durrant-Whyte: The ANSER project: Data fusion across multiple uninhabited air vehicles, Int. J. Robot. Res. **22**(7), 505–539 (2003)
[12] R.E. Moore: Interval Analysis (Prentice Hall, Upper Saddle River 1966)
[13] D. Dubois, H. Prade: Fuzzy Sets and Systems: Theory and Applications (Academic, New York 1980)
[14] S. Blackman, R. Popoli: Design and Analysis of Modern Tracking Systems (Artec House, Norwood 1999)
[15] D. Pagac, E.M. Nebot, H. Durrant-Whyte: An evidential approach to map-building for autonomous vehicles, IEEE Trans. Robot. Autom. **14**(4), 623–629 (1998)
[16] D. Hall, J. Llinas: Handbook of Multisensor Data Fusion (CRC, Boca Raton 2001)
[17] E.L. Waltz, J. Llinas: Sensor Fusion (Artec House, Norwood 1991)
[18] M. Kam, Z. Zhu, P. Kalata: Sensor fusion for mobile robot navigation, IEEE Proc. **85**, 108–119 (1997)
[19] H. Carvalho, W. Heinzelman, A. Murphy, C. Coelho: A general data fusion architecture, Proc. 6th Int. Conf. Inf. Fusion, Cairns (2003)
[20] A. Makarenko: A Decentralized Architecture for Active Sensor Networks. Ph.D. Thesis (University of Sydney, Sydney 2004)