

ELSA: A MULTISENSOR INTEGRATION ARCHITECTURE FOR INDUSTRIAL GRADING TASKS

Michael D. Naish

Computer Integrated Manufacturing Laboratory
Dept. of Mechanical and Industrial Engineering
University of Toronto
Toronto, Ontario, Canada M5S 3G8

Elizabeth A. Croft *

Industrial Automation Laboratory
Department of Mechanical Engineering
University of British Columbia
Vancouver, B.C., Canada V6T 1Z4

Abstract

This paper presents the topology of the Extended Logical Sensor Architecture (ELSA) for multisensor integration and the methodology for constructing industrial sensor integration systems based on this architecture. ELSA has been developed for industrial applications, particularly, the on-line grading and classification of non-uniform food products. It addresses a number of issues specific to industrial inspection. The system must be modular and scalable to accommodate new processes and changing customer demands. It must be easy to understand so that non-expert users can construct, modify, and maintain the system. Furthermore, a data representation scheme which allows for the quantification of product deviations from an ideal model is required.

To address these needs, the sensors are encapsulated by a logical sensor model, providing robustness and flexibility. The construction methodology is based upon the object model which represents object classifications through combinations of primary features weighted by fuzzy membership functions. The features guide the selection of sensors and processing routines; the classifications determine the rulebase used by the inference engine for process decisions. Although inspection is the focus of this work, it is intended to be applicable to a variety of automation tasks which may benefit from a multiple sensor perception system.

1. Introduction

The ability to consistently produce high-quality products is important to the success of manufacturing and processing operations. Traditional quality assurance methods have often relied on human operators who use visual cues in order to determine product quality. Such methods are tedious, time-consuming, and inconsistent. Many products, especially biologically or naturally formed items, exhibit non-standard and non-uniform characteristics. For example, products such as fish, apples, potatoes, chicken, tomatoes, and other types of produce may, even within a single classification or grade, vary widely in appearance. The problem is compounded by variations in the product characteristics within a species, region, or industry. As tolerances for acceptable products decrease, as an increasing number of varieties are offered, and as the overall demand for products increases, industry has turned to automation to address these grading and quality assurance needs.

*To whom correspondence should be addressed. Tel: 604-822-6614; fax: 604-822-2403. *E-mail address:* ecroft@mech.ubc.ca (E. A. Croft)

Machine vision systems offer a number of potential benefits to industries which rely on manual quality assurance. Since most production facilities run continuously, defects may go undetected if an inspector looks away or experiences a lapse in concentration. On the other hand, a machine vision system can guarantee that 100% of the objects leaving the system are inspected. The rate of defect detection may fall below 100% but, by increasing the inspection rate, it may achieve higher defect detection rates than a human inspector.

In addition to the reliability and repeatability of the grading system, industrial users require the ability to modify the grading scheme to meet changing market demands and customer criteria. An automated system has a consistent internal representation of product and quality classifications. This representation may be redefined by adding or removing information which governs the decision making process. Such 'global' changes offer increased consistency and flexibility over trained workers who each maintain slightly different interpretations of quality [1].

Most human multi-factor grading decisions are based on subjective interpretation of visual information and cues from other senses (e.g. smell, firmness, weight). Thus, the characterization of grading classifications is often difficult and the ability to make repeatable decisions is hampered. This problem is compounded by the nature of natural products which generally do not have crisp, ideal templates [2].

Intelligent multisensor systems are intended to provide complementary qualities to the industrial user; namely, the repeatability and reliability of automation together with the feature discrimination, classification capability, and adaptability of humans. Much of the success of machine vision and multisensor systems is dependent upon the ease of use of these systems for industrial users who may understand the process but not the details of the sensor technology. However, such systems tend to be highly industry specialized, and, at least partially, developed or contracted "in house." Therefore, it is important that the development and operation of a multisensor system is orderly, comprehensible, and simple.

The architecture and approach presented herein provides a methodology for the design and implementation of multisensor systems for industrial automation; particularly, the on-line grading and classification of non-uniform food products. Although inspection is the focus of this work, it is intended to be applicable to a variety of automation tasks which may benefit from a multiple sensor perception system. Other potential applications include material handling, assembly, and machining operations.

1.1. Multisensor Integration

Automated systems which attempt to make multi-factored decisions about non-uniform products on the basis of information from a single sensor have had limited success. Often, there is simply inadequate data for a proper product assessment. The transition to multiple sensors can extend the capabilities and improve the robustness of existing systems.

A system which employs multiple sensors may enjoy several advantages over single sensor systems [3]. For example, information can be obtained more accurately, and features undetectable with individual sensors may be perceived in less time and with less cost. These advantages are realized through the use of redundant and complementary information.

Redundant information is acquired by a group of sensors (or a single sensor over time); each sensor perceiving the same features in the environment. By integrating and/or fusing this information, the accuracy of the system can be increased by reducing the overall uncertainty. Redundant sensors also serve to increase the robustness of the system in the event of sensor failure. Complementary sensor groups, on the other hand, perceive features in the environment that are imperceptible to

individual sensors. Each sensor provides a subset of the required feature space; these feature subsets are combined to obtain the intact feature.

Little published work has been done in the area of non-specialized, sensor integration architectures for industrial applications. In the following sub-section, a brief review of general sensor integration architectures is provided.

1.2. Multisensor Integration Architectures

A system architecture provides a framework upon which individualized systems can be built and adapted. For complex systems, an architecture is essential to ensure that the system is comprehensible, robust, and that it is easily extensible. An architecture for sensor integration systems must provide the following components:

- Data structure and communication protocols.
- Resolution of information from sensors.
- Data fusion/integration engine.
- Exception handling.
- Decision making (inference from sensory information).
- Control mechanism or method of utilizing system output.

A number of different architectures have been developed for the purpose of multisensor integration, each for a specific application. For example, architectures developed for mobile robot navigation and control are primarily concerned with prioritizing objectives and ensuring that high priority (real-time) objectives are met. While, on the whole, these architectures are not directly applicable to the task of industrial inspection and classification, each of the following examples presents some aspects which are potentially useful to this problem.

An action-oriented perception paradigm is utilized by the SFX architecture developed by Murphy [4, 5]. In Murphy's architecture, sensing failures are handled by attempting to classify the error type and source using a modified generate-and-test procedure. Once the error source is identified, the error recovery module selects a predefined recovery scheme to either repair or replace the current sensing configuration. A comprehensive cognitive science model proposed by Bower [6] is used as the basis for discordance-based sensor fusion to combine information from multiple sensors. There are four fusion modes as follows:

1. Complete sensor unity. In this mode, sensor data is fused without a mechanism for detecting discordances. Sensory information is tightly coupled such that discordances do not arise.
2. Awareness of discordance where recalibration is possible. Here, the discordance between sensors is reconciled by recalibration of the offending sensors.
3. Awareness of discordance where recalibration is not possible. In this case, sensors providing erroneous data are temporarily suppressed.

4. No unity at all. Sensors observe attributes without any spatial correspondence. Here, sensor data is used independently.

Lee [7] has developed the Perception Action Network (PAN) architecture which uses modules placed along the connections between sensors to define relationships which allow the perception net to reduce uncertainties through data fusion and constraint satisfaction, in addition to identifying possible biases.

Sensor integration research within the military has focused on target tracking applications. The major issues here are proper synchronization, communication, and routing between sensor systems that are widely distributed. Architectures which have been developed include those by Iyengar *et al.* [8] based on a multilevel binary de Bruijn network (MBD) and the object-oriented approach taken by Queeney and Woods [9].

1.2.1. Logical Sensor-Based Architectures

Sensors are one of the principal building blocks of a multisensor integration architecture. The data provided by sensors may be used as input to processing algorithms which combine and convert the data into higher level representations of the information. One approach that is well suited to the incorporation of sensors into a multisensor integration architecture is the logical sensor model.

A logical sensor (LS) is an abstract definition for a sensor. Logical sensors were first defined by Henderson and Shilcrat [10] and later broadened to include a control mechanism by Henderson, Hanson, and Bhanu [11]. This definition provides a uniform framework for multisensor integration by separating physical sensors from their functional use within a system. Logical sensors are used to encapsulate both physical sensors and processing algorithms. This encapsulation defines a common interface for all sensor types allowing the straightforward addition, removal, and replacement of sensors within the architecture.

Weller, Groen, and Hertzberger adopted the logical sensor concept and developed an architecture which uses a hierarchy of sensor modules [12]. This concept was further refined by Groen, Antonissen, and Weller when applied to a model based robot vision system [13].

Dekhil and Henderson extended the concepts introduced by Weller *et al.* and introduced Instrumented Logical Sensor Systems (ILSS) [14]. The application was again mobile robot navigation. The ILSS is an extension of the LSS. The primary difference between ILSS and LSS is the addition of components which provide mechanisms for on-line monitoring and debugging.

Using the ILSS, data from physical sensors may be combined and processed using a variety of algorithms to create sensors which do not physically exist. A sensor system may be constructed which can extract complex high-level features. These features form the basis of the object representation for recognition and classification.

1.3. Industrial Applications

In the area of quality assessment and assurance, machine vision is often used to gather the bulk of the required information, especially for the grading or classification of non-uniform (biological) products. Other sensors, such as scales, mechanical measurement devices, and ultrasound are employed to gather information that is used to enhance the machine vision data. Industrial systems which employ machine vision perform one or more of the following activities [15]:

- Gauging: Performing precise dimensional measurements.
- Verification: Ensuring that one or more desired features are present and/or undesired features are absent.
- Flaw detection: Location and segmentation of undesired features which may be of unknown size, location, and shape.
- Identification: Use of symbols, including alphanumeric characters, to determine the identity of an object.
- Recognition: Use of observed features to determine the identity of an object.
- Locating: Determination of object location and orientation.

There have been a number of vision-based multisensor systems developed for quality assurance and assessment over the past decade. In most of these applications, ad-hoc methods are used to develop a sensor integration system to monitor the process. Such systems lack a formal architecture and are typically designed by experts in machine vision and/or systems integration. This can result in difficulties with the use and maintenance of the system for the everyday user. Additionally, upgrading the system to change or add additional sensors and/or requirements often requires the system to be redesigned. This is a problem for industrial users whose requirements in terms of speed, feature recognition, accuracy, and other process monitoring parameters invariably change over time. A number of examples of recent industrial systems follow.

Daley *et al.* [16] are working towards the automation of poultry grading and inspection. This system uses a colour CCD camera to obtain information regarding the HSI colour, size, and shape of each bird. Global defects are identified with a 96% success rate; the difficulty of extracting local defects limits success to about 60%. To properly address the problem, additional sensors are required to allow for the measurement of the surface texture and structure.

A low-cost system for fruit and vegetable grading was developed by Calpe *et al.* [17] as an alternative to expensive commercial systems. This open platform may handle up to 12 lanes simultaneously at a speed of 10 items per second. Classification is based on the average of four RGB colour images of each piece of produce.

Recent work in the Industrial Automation Laboratory at the University of British Columbia has involved the grading of herring roe skeins [18, 19]. Sensors provide information regarding colour, contour, weight, and firmness. All of this information is combined to determine a classification for each roe. Classification accuracy between Grade 1 and Grade 2 roe is about 95%; however, additional sensors are required to improve the subclassification of the Grade 2 roe into various sub-grades.

Other applications which make use of sensory information for grading and classification include potato grading [20], shrimp inspection [21], printed circuit board inspection [22], and visual inspection of unsealed canned salmon [23].

A number of proprietary industrial systems exist for product inspection and classification. These include the QualiVision system from Dipix Technologies Inc. for the quality control of bakery and snack food products. This system uses 3D imaging to assess product consistency to 10 microns [24]. Lumetech A/S has developed the Fisheye Waterjet Portion Cutter for trimming and portioning fish fillets [25]. Key Technologies Inc. offers the Tegra system for grading agricultural products according to size and colour [26]. Typically, such systems sort products based on 1–2 discrete thresholds.

1.4. Objectives

To address the industrial needs outlined above, a new, open architecture approach for intelligent multisensor integration in an industrial environment is proposed. The objectives of this architecture and methodology are as follows:

1. To provide a modular and scalable architecture which serves as a robust platform for intelligent industrial sensing applications.
2. To specify an encapsulation of physical devices and processing algorithms.
3. To specify a data representation scheme which allows for the quantification of deviations from an ideal model.
4. To ensure that the data representation scheme provides the user with insight as to how the system is structured and how the sensor information is used to make decisions.
5. To provide a robust exception handling mechanism to ensure the reliability of an implementation of this architecture.
6. To ensure that the architecture is applicable to a broad range of industrial applications, especially those involving non-uniform product grading.

Herein, objectives 1, 2, 5, and 6 are addressed by the Extended Logical Sensor Architecture (ELSA). A brief review of the object model (objectives 3 and 4) is provided; further details may be found in [27].

2. System Architecture

This section presents the basic structure of the Extended Logical Sensor Architecture for multisensor integration. The ELSA architecture may be decomposed into three groups, according to the following tasks:

1. **Sensing:** The acquisition of information from the environment which is used as the basis for inference and decision making.
2. **Inference:** The combination of the sensory information with information contained in a knowledge base to infer decisions.
3. **Action:** The conversion of decisions into commands and signals which control process machinery.

The structure of ELSA is illustrated in Figure 1. An object-oriented approach to system configuration has been adopted. The encapsulation of the primary components leads to a scalable and flexible system which is particularly suited to industrial grading tasks. The system may be easily reconfigured to adapt to advances in sensor and processing technologies or changing market demands. Due to the nature of industrial inspection and grading, the primary focus of this work is on the sensing and inference groups.

Sensing is performed by the coordinated actions of the sensors, the Integration Controller, and the Validation and Diagnostic modules. Sensors are encapsulated by a logical sensor model. The Integration Controller is capable of coordinating the reconfiguration of the sensor hierarchy to meet process goals. This is assisted by knowledge contained in the Knowledge Base which is shared with the Inference Engine.

Process decisions are made by the Inference Engine. The validated sensor information from the sensing group provides the required input to the Rulebase. The action group includes the Post Processor, drivers, and process machinery. Control systems for grading systems typically range from very simple to extremely complex. Herein, the details of the control issues associated with the action group are not considered and are open problems for future work.

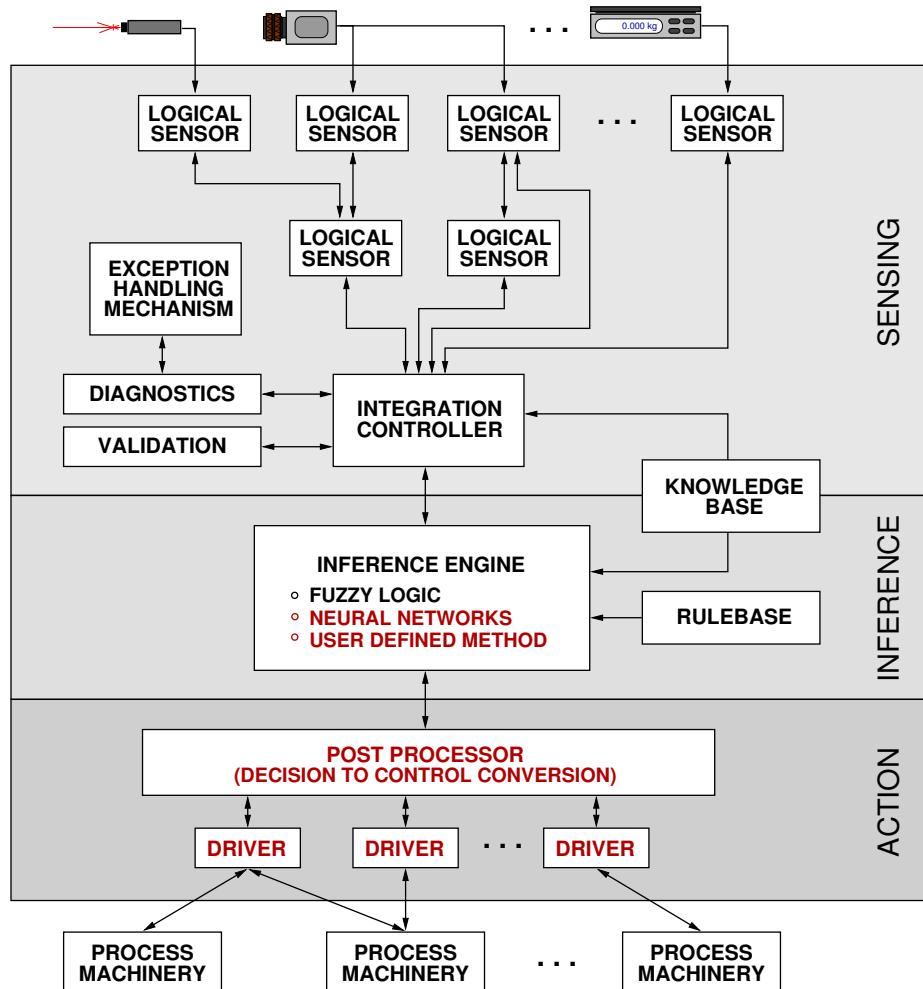


Fig. 1: Overview of Extended Logical Sensor Architecture.

2.1. Logical Sensors

The logical sensor hierarchy structures data in a bottom-up manner. The raw data collected by the physical sensors is processed through different levels of logical sensors to produce high-level representations of sensed objects and features. This approach offers considerable flexibility. High-level tasks may be implemented without regard to the specific sensing devices. The low-level physical sensors and low-level data processing routines are invisible to the higher levels. That is, to higher-level sensors, each antecedent (lower-level) logical sensor appears as a single entity with a single output, regardless of the scope of *its* antecedents. Using the logical sensor model, a hierarchy of subordinate and controlling sensors can be built, ultimately providing sensor input to the Integration Controller.

The logical sensor model outlined in Section 1.2.1. is extended herein for a model-driven open architecture. As shown in Figure 2, the proposed Extended Logical Sensor (ELS) is comprised of a number of different components. The components are object-oriented by design; each component is responsible for a single task within the sensor. A list of these components and tasks is given in Table 1. The ELS strongly encapsulates the internal workings of each logical sensor while allowing for the modification of the sensor's operating characteristics. The components of this revised model are outlined in greater detail

in the sections or references provided in the last column of Table 1.

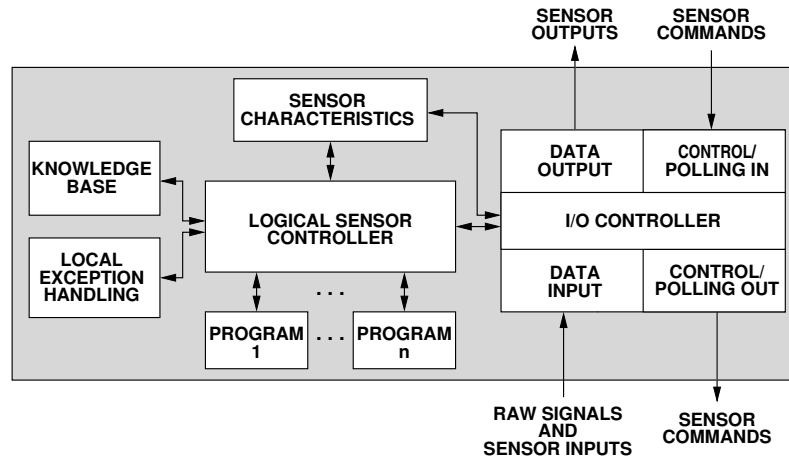


Fig. 2: Basic components of an Extended Logical Sensor.

The control command mechanism is flexible enough to accommodate active sensors; for example, a camera in an active vision system may be repositioned to bring an object of interest into (better) view. However, since the target applications are industrial in nature, namely, inspection and grading tasks, herein the sensors are assumed to be passive.

As will become apparent, the implementation of an ELS requires an understanding of signal processing. This is knowledge that most industrial users will not possess. They will understand *what* they would like the ELS to do, but not necessarily *how* to accomplish it. This limitation is overcome to some degree by the development and provision of an ELS library which contains a variety of logical sensors for many common signal processing operations. When a required ELS is not available in the library, it will be necessary to have others implement it.

2.1.1. Logical Sensor Characteristics

The logical sensor characteristics refer to a set of properties specific to each logical sensor (LS). This information is publicly accessible, enabling other logical sensors, or the Integration Controller, to poll the sensor and determine the sensor's identity and capabilities. The components which comprise the logical sensor characteristics are: the Logical Sensor Name, the Characteristic Output Vector, the Sensor Function, and the Sensor Dependency List. The first two characteristics were defined by Henderson and Shilcrat [10]; the other characteristics are new, and are described below.

The Sensor Function provides a description of functionality of the logical sensor. This description is in human readable form so that a user may effectively browse through a library of logical sensors. As an example, an edge detection ELS would have a description indicating that it was capable of identifying sets of edge pixels from a two-dimensional array of pixel intensity values. In addition, comments on accuracy and computational complexity (speed and memory requirements) would assist the user and the system in comparing this edge detector with others which may be available. This information may then be used to select the most appropriate edge detector for a given task.

The Sensor Dependency List provides a list of the logical sensors subordinate to the ELS being polled. Each ELS which provides input to one of the logical sensor programs is considered as a subordinate. An ELS is identified by its Logical Sensor

Table 1: Summary of Extended Logical Sensor components.

Component Group	Component	Description	Reference
Sensor Characteristics	Logical Sensor Name	Uniquely identifies a particular logical sensor to the system. By definition, a name may not be duplicated within the hierarchy. Similar sensors are numbered consecutively.	Henderson and Shilcrat [10].
	Characteristic Output Vector	A vector of types which serves to define the output vectors that will be produced by the logical sensor.	Henderson and Shilcrat [10].
	Sensor Function	A description of the functionality that this sensor provides. Provided in human readable form.	Section 2.1.1.
	Sensor Dependency List	A list of dependencies for the logical sensor, accounting for each logical sensor that serves as input to the contained programs.	Section 2.1.1.
I/O	I/O Controller	Monitors, redirects, and packages data and control commands for inter-sensor communication.	Section 2.1.2.; and, Henderson et al. [11]
	Data Input	Consists of signals from transducers and data from logical sensors.	Section 2.1.2.
	Data Output	Output in the form of the characteristic output vector, error messages, or polling results.	Section 2.1.2.
	Control Input	Interprets the control structure used for commanding and adjusting sensors for changing conditions and goals.	Section 2.1.2.; and, Dekhil and Henderson [14].
	Control Output	Control commands to subordinate sensors. May be generated by sensor or passed through from higher level sensors.	Section 2.1.2.
Controller	Logical Sensor Controller	Acts as a “micro”expert system to ensure the optimal performance of the logical sensor.	Section 2.1.3.; and, Henderson and Shilcrat [10].
	Local Exception Handling	Internal diagnostics and error handling. Works in conjunction with logical sensor controller. Attempts to classify the error and then rectify the problem using a predefined recovery scheme.	Section 2.1.3.; and, Dekhil and Henderson [14].
	Local Knowledge Base	Contains information on interpretation of control commands for adjustment of parameters and selection of programs. Also stores default parameters used during initialization and reset.	Section 2.1.3.
Programs	Device Drivers	Used to interpret raw signals from physical sensory devices.	Section 2.1.4.
	Processing Algorithms	Signal processing routines used to extract features and information from sensor data.	Henderson and Shilcrat [10].

Name. This list is automatically generated as the ELS hierarchy is constructed.

2.1.2. I/O

The I/O Controller is an extension of the Control Command Interpreter [11], that provides a specification for control to the original logical sensor specification [10]. The I/O Controller oversees all inputs and outputs from the ELS and monitors, redirects, and packages data and control commands for inter-sensor communication. For control commands, the controller works as a pass-through buffer. The destination logical sensor name of each control object received by the I/O Controller is first checked to determine if the command is intended for the particular sensor. If so, the control command is interpreted and sent to the LS Controller for processing; if not, it is passed through to lower-level (subordinate) sensors.

One can note that higher-level sensors may only be aware of the *function* of each subordinate ELS. The details of the actual algorithms — and, in the case of sensors with multiple programs, the currently selected algorithm — is hidden from

higher-level sensors by encapsulation. As a result, commands (and associated parameters) generally request a desired *effect*. For example, a command to increase the number of edges extracted from an array of pixel intensities would be of the form INCREASE EDGES. The specific algorithm used need not be known. This command would be passed down through the hierarchy to the edge detecting ELS. At this sensor, the controller would interpret this command and, drawing upon information contained in the Local Knowledge Base, adjust specific algorithm parameters accordingly (such as reducing mask size or threshold values).

Data sources for an ELS may be either raw signals from physical transducers in digital form (to be handled by an appropriate software driver), or logical sensor data in the form of the Characteristic Output Vector (COV). To properly interpret data from subordinate sensors, the I/O controller must have an internal copy of the COV for each connected lower-level ELS. This internal copy is obtained through sensor polling.

The data output module serves to package the data from an ELS program into the form of the COV, a polling result in response to a query, or, in the case of failure, an error message. Failure of a ELS may occur due to the failure of a lower-level ELS or an inadequacy of a contained algorithm. In either case, the confidence measure which accompanies each ELS output will fall below a specified tolerance. An error message will then be passed in place of the output vector.

The confidence measure is generated by the ELS. In the case of an encapsulated physical sensor, the uncertainty measure is based upon the specifications and/or known operational characteristics of the device. Algorithms within the ELS must provide routines which calculate the uncertainty associated with each output value. Confidence is represented as a real-valued number in the range: $0 < c < 1$. A measure near 0 indicates little confidence in the result; while a measure near 1 indicates a high level of confidence in the sensor output.

The ELS provides a control structure which allows for the adjustment of logical sensors in response to changing conditions. Each command is packaged as a control object which identifies the destination ELS(s), specifies the control command, and any associated parameters. Control output from an ELS is sent to lower-level logical sensors. These may be generated by the issuing sensor, or may be passed through from a logical sensor at a higher level.

2.1.3. Controller

The controller is comprised of three components which work together to supervise the internal operation of the ELS. The first component is the Logical Sensor Controller which is similar to the Selector of the original LSS in that it monitors the output from each program; however, a Local Knowledge Base and Exception Handling Mechanism are used to increase functionality while maintaining the encapsulation of the sensor operation.

The LS Controller provides the logical sensor with a mechanism to respond to commands passed from the I/O Controller. A number of standard control commands are defined for all logical sensors, namely, commands used for sensor initialization, calibration, requests for sensing, testing, and reconfiguration. These, in addition to user commands, are stored locally for each ELS. A copy of user commands is also maintained by the Integration Controller. This provides controlling sensors with information about the capabilities of subordinate sensors.

The second component is the Local Exception Handling module. It is responsible for internal diagnostics, local error detection, and recovery. The testing and recovery schemes are limited to the domain of the ELS, using the methodology outlined in Section 2.2.1.. Errors which cannot be handled locally result in the sensor issuing an error message. Typically,

these errors are passed to the Integration Controller, which attempts to rectify the problem from a global, rather than local, perspective.

The third component, namely, the Local Knowledge Base contains a variety of information which is essential to the operation of the ELS. Among the information contained in the Knowledge Base are default parameters used during initialization and reset; command definitions, both local and standard; operating characteristics used for program parameter adjustment; criteria for monitoring sensor performance; tests to determine error causes; local error definitions for sensor specific problems; and error mappings which are used to assist in error recovery. In general, this information is not available to other sensors or modules in the system.

2.1.4. Programs

Each ELS must contain at least one program to process the input data; however, when possible, each logical sensor may contain a number of alternate programs. There are two main reasons that multiple programs may be desirable within a LS:

1. Multiple programs enable the use of combinations of different input sources.
2. Different algorithms may be used to process the input data at different rates or with different degrees of precision. This provides a mechanism for *sensor granularity*. For example, a high-speed, coarse interpretation may be used in place of a low-speed, high-resolution interpretation in time-critical situations.

The performance of the ELS is affected by the selected program and the adjustment of the program parameters. An alternate program may be selected in response to a sensor failure or in response to a command passed from a controlling sensor. In the case of a sensor failure, the alternate program selected typically relies on an alternate set of logical sensors for input. This redundancy provides a measure of robustness to the sensor system.

While the method of data generation may be different for each program within the ELS, each must be capable of providing data in the format specified by the COV. Programs may be either device drivers or processing algorithms. Device drivers are used only for direct interaction with physical transducers. Each physical device has an associated driver which, in addition to signal interpretation, manages the actual data transfer and control operations. Processing algorithms perform signal processing tasks, extracting the information required for the task at hand. Should sensor fusion be desirable for a particular application, it is performed by an ELS that is selected or designed for this task. Any fusion mechanism may be employed, though the discordance-based sensor fusion method presented by Murphy [5] is used herein for its robustness.

2.2. Integration

Integration involves the packaging of the sensory information provided by the logical sensors into a form suitable for the Inference Engine. Extracted information and features from top-level logical sensors are used to provide high-level representations of the objects of interest. As this is the final stage before decisions are made based on the sensor data, particular attention is paid to ensure data integrity.

The Integration Controller oversees the operation of the system, acting as an interface between the top-level logical sensors and the Inference Engine. Here, the concept of what the system is trying to accomplish is maintained. It serves to coordinate sensor integration, in addition to data validation and exception handling activities which cannot be handled at the ELS level.

Sensor uncertainty is used throughout the integration process. Confidence measures are used for the identification of sensing errors and for the integration of sensor data. Sensor performance criteria are maintained in the system Knowledge Base. These criteria are used to determine whether the data provided by the sensors lies within acceptable ranges or is of an expected form. All data which is successfully validated is passed to the Inference Engine; problematic data is passed to the Diagnostics module.

2.2.1. *Exception Handling*

Exception handling aims to maintain the successful operation of the system in the event of sensor failure. Exception handling routines are invoked when data fails to satisfy a predetermined constraint or is in conflict with data from another sensor. Exceptions are handled by first classifying the nature of the error. Once classified, an attempt is made to rectify the cause of the error.

Without the availability of a complete causal model, detected errors must be classified so that the appropriate corrective action may be taken. Sensor failures are classified into three types as follows:

1. Sensor malfunctions: This occurs when one or more sensors are malfunctioning. Examples include power failure, impact damage, miscalibration, etc.
2. Environmental change: One or more sensors are not performing properly because the environmental conditions have changed since sensor configuration and calibration.
3. Errant expectation: Sensor performance is poor because the sought object is occluded or lies outside of the sensor's "field of view."

Error classification is accomplished by a generate-and-test algorithm [28]. The suspect sensors are first identified. An ordered list of possible hypotheses explaining the sensor failure is then generated. Each hypothesis is associated with a test which may be used for verification. These tests are performed in an effort to confirm or deny the proposed hypotheses. This process is repeated until a hypothesis is confirmed.

The generate and test method does not require formal operators for the generation of hypotheses. This allows the system to use a rule-based method to select from a list of candidate hypotheses. Unfortunately, this method can be time consuming if there is a large problem space and all hypotheses must be generated. This disadvantage may be overcome by constraining the problem space, thereby limiting the number of hypotheses and reducing processing time. Testing is conducted until all tests have been performed or an environmental change has been detected. When the classifier is unable to resolve the cause of the error, the cause is assumed to be an errant expectation.

For each error cause, there would ideally be a number of different recovery schemes. From these, the most appropriate would be selected by the Exception Handling Mechanism. To limit the scope of the problem and reduce the overall recovery time, a direct one-to-one mapping of error causes to recovery schemes is utilized. A library of cases allows for the instant mapping of error cause to recovery scheme based on the error classification.

Functions are used to repair individual sensors or reconfigure the sensor hierarchy. The sensor parameters are adjusted first — recalibration is accomplished by invoking a predefined sensor calibration routine. If the sensing configuration cannot

be repaired through parameter adjustment or recalibration, the sensor hierarchy is altered. The alteration may suppress a particular sensor or remove sensors from the hierarchy.

2.3. Inference Engine

Once the sensory information collected by the logical sensors has been validated, it is passed to the Inference Engine. Here, based upon the examination of the extracted objects and features, decisions are made regarding the actions to be taken with each object.

The sensor inputs are used to form the antecedents of the control decisions to be made in the Inference Engine. The consequents of these rules are the actual decisions. These are passed from the Inference Engine to the Post Processor for conversion into action.

The Inference Engine is designed to make use of cognitive-based models such as fuzzy logic [29] and knowledge based systems selected and configured by the user. These models are advantageous in that they allow the incorporation of expert domain knowledge. This expert knowledge may be formulated into a rulebase to serve as the basis of fuzzy inference.

For applications where expert knowledge is less concrete, a feature-based inference technique such as artificial neural networks [30] may be used to interpret the sensor information and produce control decisions. In this case, the network must be interactively trained to produce the desired results. Other possibilities for feature-based inference techniques include Bayesian reasoning and the Dempster-Shafer theory of evidence.

2.3.1. Rule/Knowledge Base

The Knowledge Base is utilized by both the Integration Controller and the Inference Engine. Each requires different information. The Integration Controller relies on the Knowledge Base for information about the characteristics of various logical sensors, error classifications, error recovery schemes, and other information necessary for the operation and coordination of the sensors.

The Inference Engine uses the Knowledge Base as a repository of domain knowledge. For fuzzy logic or knowledge based inference this consists of expert domain knowledge to be supplied by the user. For grading and inspection tasks in particular, the expert knowledge available from human inspectors is available to the system designers. This information is stored as a set of rules and membership functions which form the antecedents and consequents for decision making. Contextual knowledge may also be used to weight the contribution of various sensors.

2.4. Post Processing

Once the Inference Engine has processed the sensory information and interpreted it, any decisions made must be converted to actions. This involves the conversion of a directive into a plan of action for execution. The Post Processor acts as an interface between the Inference Engine and the drivers which are used to control the process machinery. Drivers are then used to convert control actions from the controller into the specific format required by each device. Devices may range from simple actuators such as solenoids and electromagnets, to complex systems such as multiple degree of freedom robotic manipulators.

3. Object Modelling

The object model is intended to provide the user with an intuitive understanding of how to construct and represent real-world objects. The model provides a basis by which the system is constructed and configured without requiring the user to possess an in-depth knowledge of the system's internal workings.

Given a data set provided by a set of sensors to perform some level of object recognition, an object model is used to classify the data. The model provides a generalized description of each object to be recognized. For industrial grading applications, the object model must represent the important features which designate the 'grade' or value of a particular object. Ideally, the model is simple to construct.

There are many methodologies for object recognition and representation; however much of the research in the field has focused on the recognition of generic objects, categorizing objects into broad groupings. With the exception of facial and handwriting recognition, little work has been done to develop systems capable of detecting subtle differences. This is the requirement of an industrial inspection and grading system where the detection and classification of subtle differences is required. The problem is not one of differentiating an apple from an orange, but rather one of discriminating the quality of a particular apple based on such cues as colour, size, weight, surface texture, and shape.

ELSA utilizes feature-based recognition, using distinguishing features to recognize and differentiate objects without discriminating every feature of each object. This uses the theory of recognition-by-components (RBC) proposed by Biederman [31] which suggests that objects are recognized, and may therefore be represented, by a small number of simple components and the relations between them. A brief review of ELSA's object model [27] is provided herein.

Objects are represented by a connected graph structure, Figure 3, similar to that proposed by Tomita and Tsuji [32] for the recognition of objects from texture features. The object model, in contrast to the logical sensor hierarchy, is a top-down representation of an object. Object nodes represent salient features of an object. The object itself is represented by the classification layer, which is the highest level of abstraction. An object whose relevant features are invariant or which does not require classification may have a single node in the classification layer.

Below the classification layer lie nodes representing the high-level features upon which classifications are made. Traversing down the graph, further into the feature layer, other nodes represent the mid and low-level features of the object. Each subsequent level becomes more and more detailed. This enables compact and efficient object models. Only the level of detail required for identification or classification need be specified.

This approach allows for scalable complexity of the object model. The hierarchical structure minimizes the disturbance to the model should a feature used for classification require modification. This is similar to the encapsulation provided by the logical sensors — refinement may focus on specific features and classifications without disturbing the others.

Fuzzy logic provides a mechanism by which human expertise may be applied in a form very close to our natural language [33]. ELSA utilizes fuzzy links to relate object features with fuzzy membership functions. This enables the system to incorporate human expertise for the determination of object classifications.

Classification is achieved using the compositional rule of inference to combine each of the primary features, yielding a total classification score. The descriptor associated with the fuzzy link serves to weight the contribution of each feature to a particular classification. The classification with the highest total 'score' is chosen. For example, if a primary features closely

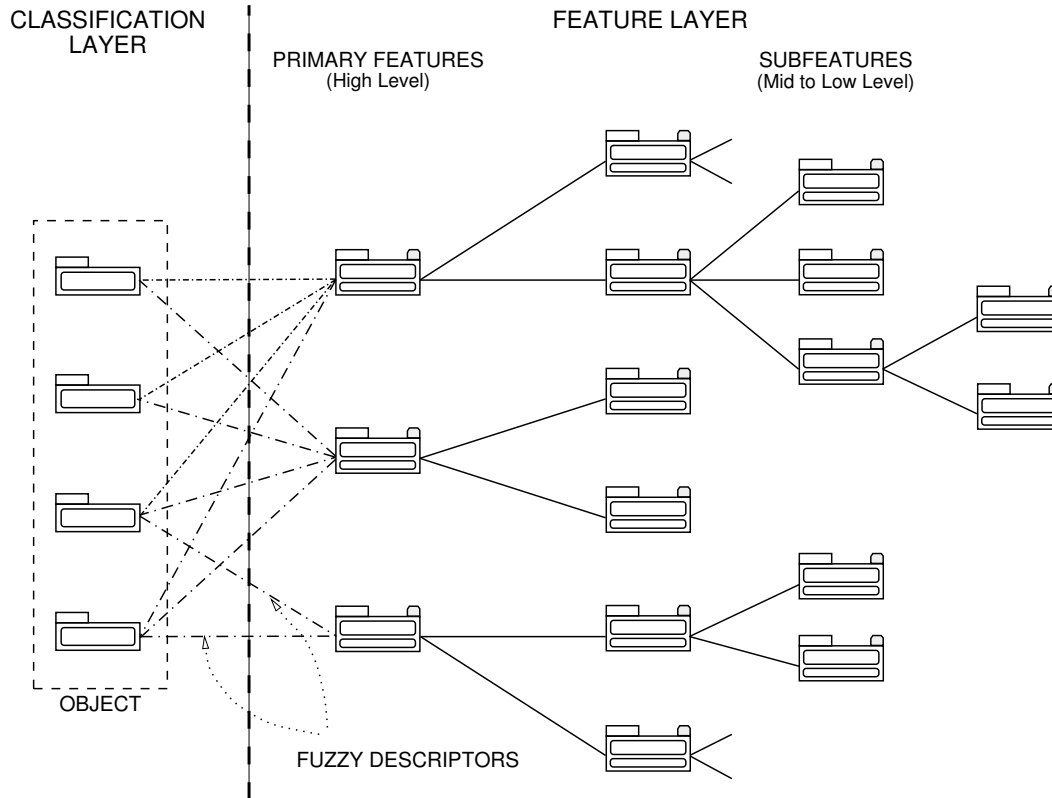


Fig. 3: Graph structure for object representation.

matches the fuzzy descriptors for a classification, its 'score' is near 1.

This structure satisfies two objectives. The first is to provide a representation for real-world objects which allows for the quantification of deviations from an ideal model. Secondly, the structure serves to guide the user through the development of the object model. Once created, the modelled features provide a basis for the selection of the logical sensors which will provide the required information to the system. The modelling of object classifications serves as a basis for the development of the Inference Engine.

4. Construction Methodology

To maximize system robustness and usability, the construction of an industrial sensing and processing system using ELSA follows a set procedure. An overview of this methodology is presented in Figure 4. The sub-sections that follow detail the various phases of the process. The methodology will be further illustrated by the example provided in Section 5.

4.1. Problem Definition/Requirements Specification

The first phase of the design process involves the recognition of the needs of the particular industry or process. From these needs, a clear statement of the problem to be solved may be formulated. This problem definition is more specific than the general needs; it must include all of the specifications for what is to be designed. Hence, the designer must consider the required system capabilities. Following the general principles for system design outlined in [34], a set of minimum functional

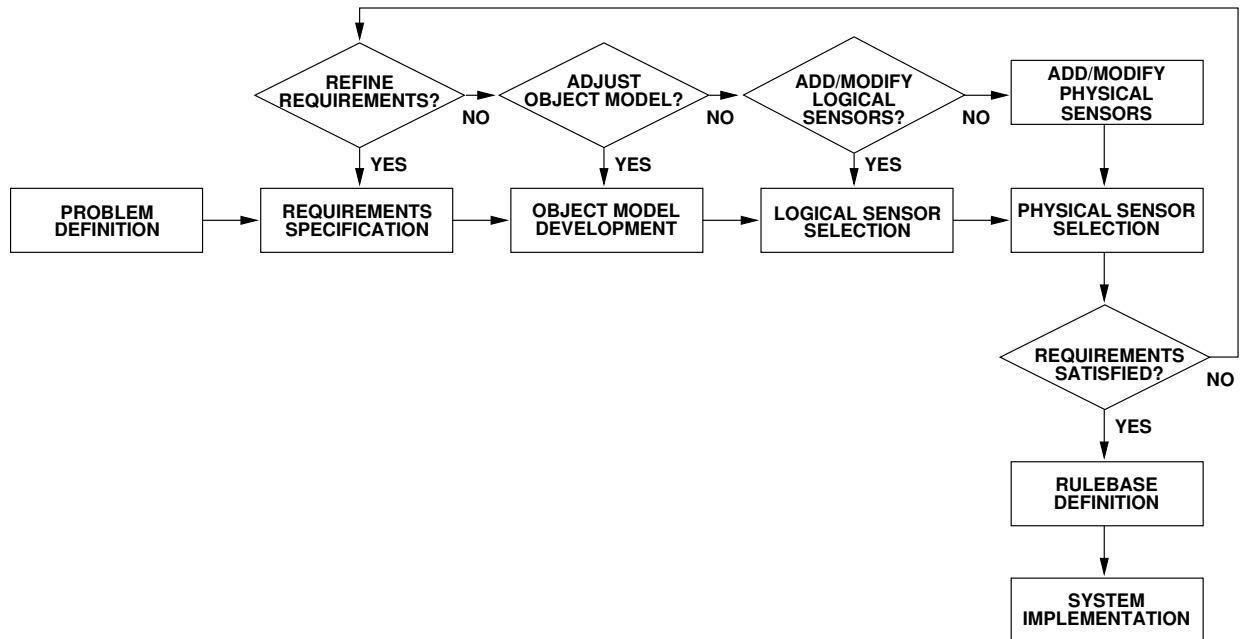


Fig. 4: Overview of construction methodology.

requirements is specified. The articulation of these requirements is used as a guide for subsequent phases. If any of the requirements are left unsatisfied, the design is inadequate. The requirements also serve to keep the design focused on what is necessary for the task at hand.

4.2. Object Model Development

Object model development for ELSA is a two-stage process. First, based on the system requirements from the previous phase, the primary features or characteristics upon which classifications are to be made are identified. As discussed in Section 3, it is advantageous to keep the size of this set to a minimum. Typically, the features in this set are at a high level of abstraction. They occupy the top of the feature layer of the model (Figure 3). From this set, each feature which is not atomic is decomposed into a set of subfeatures. Thus decomposition continues until all features are atomic. A feature is considered to be atomic if it cannot be subdivided further. This process is illustrated in the left-half of the flowchart in Figure 5.

Once high-level features are represented by atomic features in the lower section of the object model, this high-level information is used to define the object classifications following the steps in the right-half of Figure 5. The classifications occupy the upper level of the model topology (left side of Figure 3). Fuzzy links combine the high-level feature information into object classifications. By adjusting the combination of features and their corresponding fuzzy descriptors, object classifications may be distinguished. The classification layer of the object model (relevant features in combination with relative weights) serves as a template for the inference engine which, in practice, makes the classification decisions based on the feature information extracted by the logical sensors.

The object model may be refined by adjusting the classifications and/or the primary (and subordinate) features. When an object is improperly classified, the fuzzy descriptors associated with each fuzzy link to the classification may be adjusted

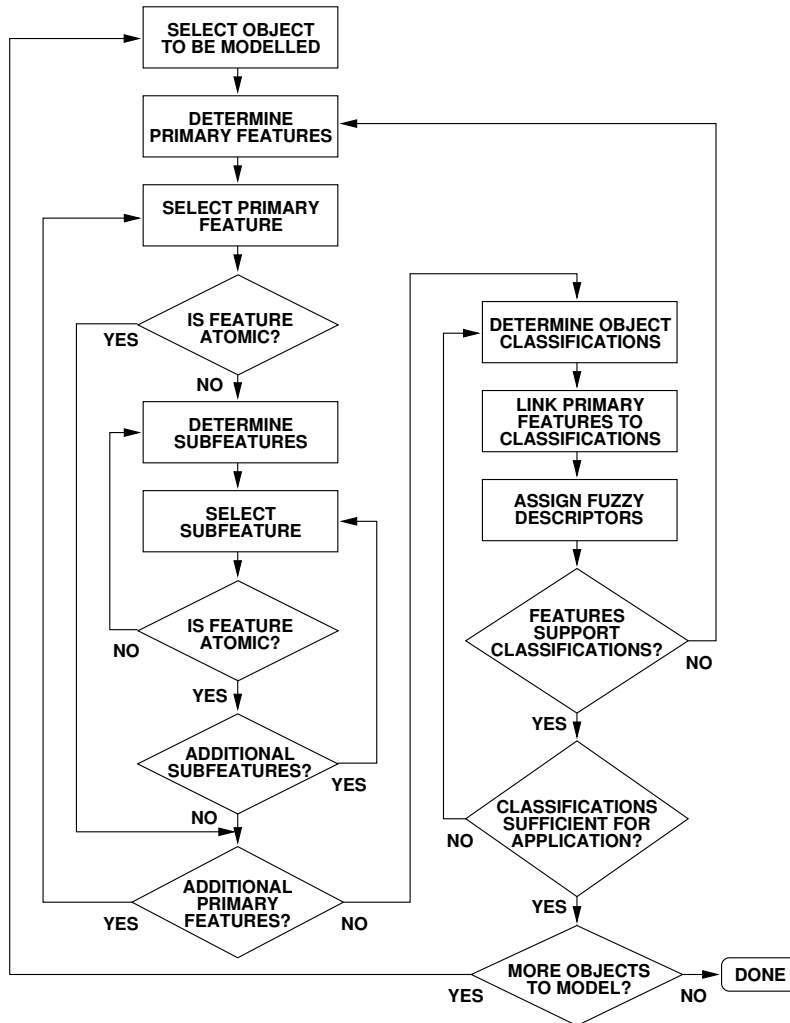


Fig. 5: Methodology for development of object models.

to rectify the problem. Should this be unsuccessful, it may be necessary to include or define an additional primary feature. Problems with feature extraction may be handled through parameter adjustment and the refinement of properties and relations.

4.3. Logical/Physical Sensor Selection

The selection of logical sensors is driven by the primary, intermediate, and atomic features that have been identified as necessary for the object model. Sensor selection starts with the primary features. Each feature has a corresponding ELS which packages the information from lower-level sensors (logical or physical) into the representations used for object classification. Many of the low-level logical sensors are selected from a reusable ELS library. The logical sensors contained within the library perform standard image and signal processing operations. The algorithm for constructing the ELS hierarchy is given in Figure 6.

Physical sensors are selected to satisfy the input requirements of the ELS associated with each atomic feature. This requires a consideration of both the input requirements and the capabilities of available transducers. A feature that is beyond

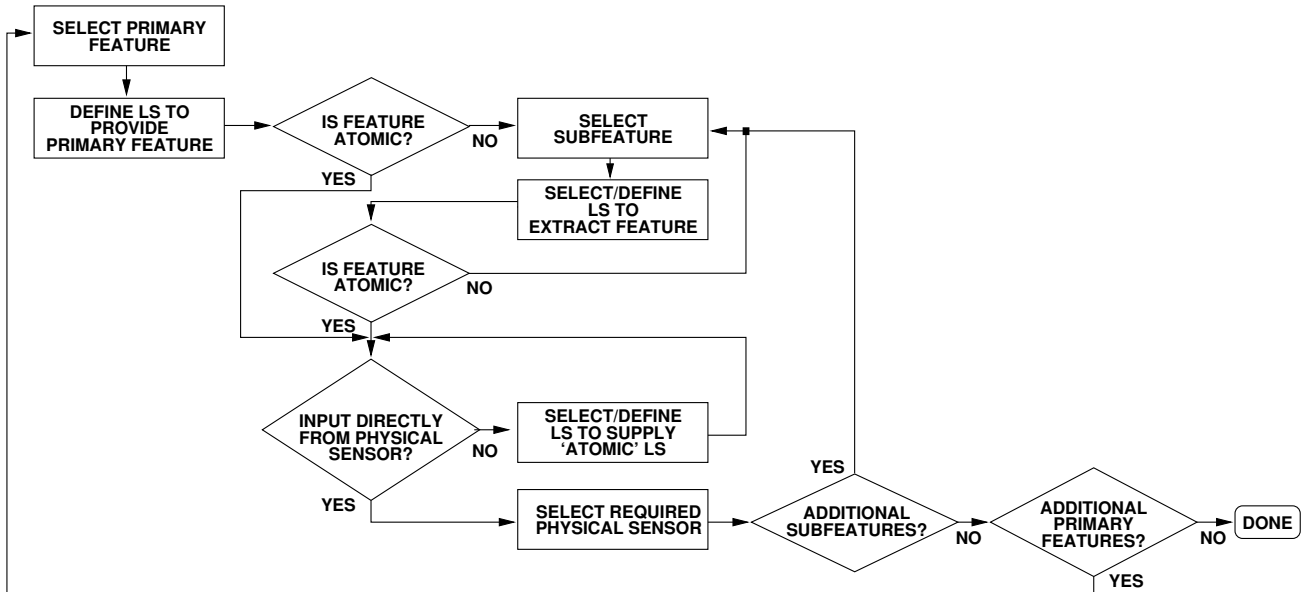


Fig. 6: Methodology for the development of the ELS hierarchy.

the range or capabilities of a single sensor may be accommodated by the fusion of data from multiple sensors which cover the feature space. An ELS is then defined which provides the feature, fusing the data from each of the physical sensor inputs.

Other considerations include whether the system should attempt to utilize a single sensor for multiple tasks or whether specialized sensors will be used. For example, a camera can provide size, colour, and shape information. Clearly, separate cameras are not required to extract each of these features. Using visual information and a correlation between length, area, and mass, a weight ELS may be defined to estimate the weight of an object. Depending on the application, this may be used to replace or augment the information from a load cell.

4.4. Rulebase Definition

The Rulebase defines both rules for object classification and rules to infer the appropriate system output from these classifications. It is generated directly from the object classifications contained in the object model.

The classification rules use the fuzzy descriptions of each classification as the basis for description. The confidence in the detection of each primary feature may then be used as input to the classification rules. Each rule expresses a degree of certainty in the classification of the object based on the detection of the primary features. The rules for each classification are combined using the compositional rule of inference to produce the certainty that the object is of each classification.

Decision rules are defined to inform the system what should be done according to how each object is classified. Decisions are defined using the confidence in each object classification as the antecedent(s); the appropriate decision(s) forms the consequent. If an object classification is certain, the appropriate decision is straightforward. By evaluating the confidence of each object classification, borderline cases may be handled in the most appropriate manner.

The membership functions required for the definition of these rules are generated automatically so that users are not required to have an in-depth understanding of fuzzy logic. Classification of features which are not easily quantified use the

membership functions *low*, *average*, and *high* to express the confidence in the detection of the feature. These functions span the range 0 to 1; *average* is centred over 0.5. This is intended to provide users with an intuitive feel for the specification of classifications. The user does not consider values or fuzzy membership, but rather the linguistic variables *low*, *average*, and *high*.

For features that are easily quantified, such as length and mass, the system prompts the user to supply information about the universe of discourse (range of expected values), and linguistic variables for the classifications over this universe. Triangular membership functions centred at the mean values of each variable are used, since with sufficient representation the membership function shape is not critical [29]. Expert users may by-pass the system, allowing direct definition and fine-tuning of the membership functions.

Inferring action from the object classifications uses a similar methodology. The membership functions *uncertain*, *unsure*, and *certain* are used to span the universe of certainty. Again these functions are defined over the range 0 to 1, with *unsure* centred over 0.5.

5. Application Example

The following sub-sections present an example of system construction based on ELSA. This example does not create or demonstrate an automated industrial prototype, but rather serves to illustrate how the ELSA methodology could be used to construct a sensor integration system for product inspection. The object model, ELS hierarchy, and Inference Engine are developed using the ELSA approach.

5.1. Background

Herring roe is an important part of the B.C. economy, with an annual value of \$200 million dollars. A herring roe skein is a sac of tiny herring eggs. Being a natural product, it exhibits many non-uniform characteristics. Roe is a particularly challenging product due the large number of classifications. Each classification is dependent on the presence or absence of a number of features. Appearance and texture of the salted herring roe are the primary factors influencing price. Proper classification allows processors to offer improved value to their customers.

Currently, the process of grade classification is done manually. Herring roe is assigned a subjective grade according to aesthetic properties including colour, texture, size, and shape. Of these, all but texture are assessed visually; texture is assessed by tactile examination.

The roe grades are subject to change each season, due to the customer driven nature of the industry. Currently, there is no standardization of the various grade specifications. Distortions of the roe are commonly described using linguistic terms — the interpretation of which varies among expert graders. This inconsistency makes the quantification of product quality difficult. Examples of good (Grade 1) and distorted (Grade 2 and 2-H) roe are presented in Figure 7.

5.2. Problem Definition/Requirements Specification

The shape of the roe has been the most difficult to access using machine vision [18]. Human graders look for roe to be ‘well formed,’ or free from defects. The current industrial prototype [19], limited to the use of a single image without intensity information, is unable to consistently classify roe with various defects. The original system was designed as a two-classifier

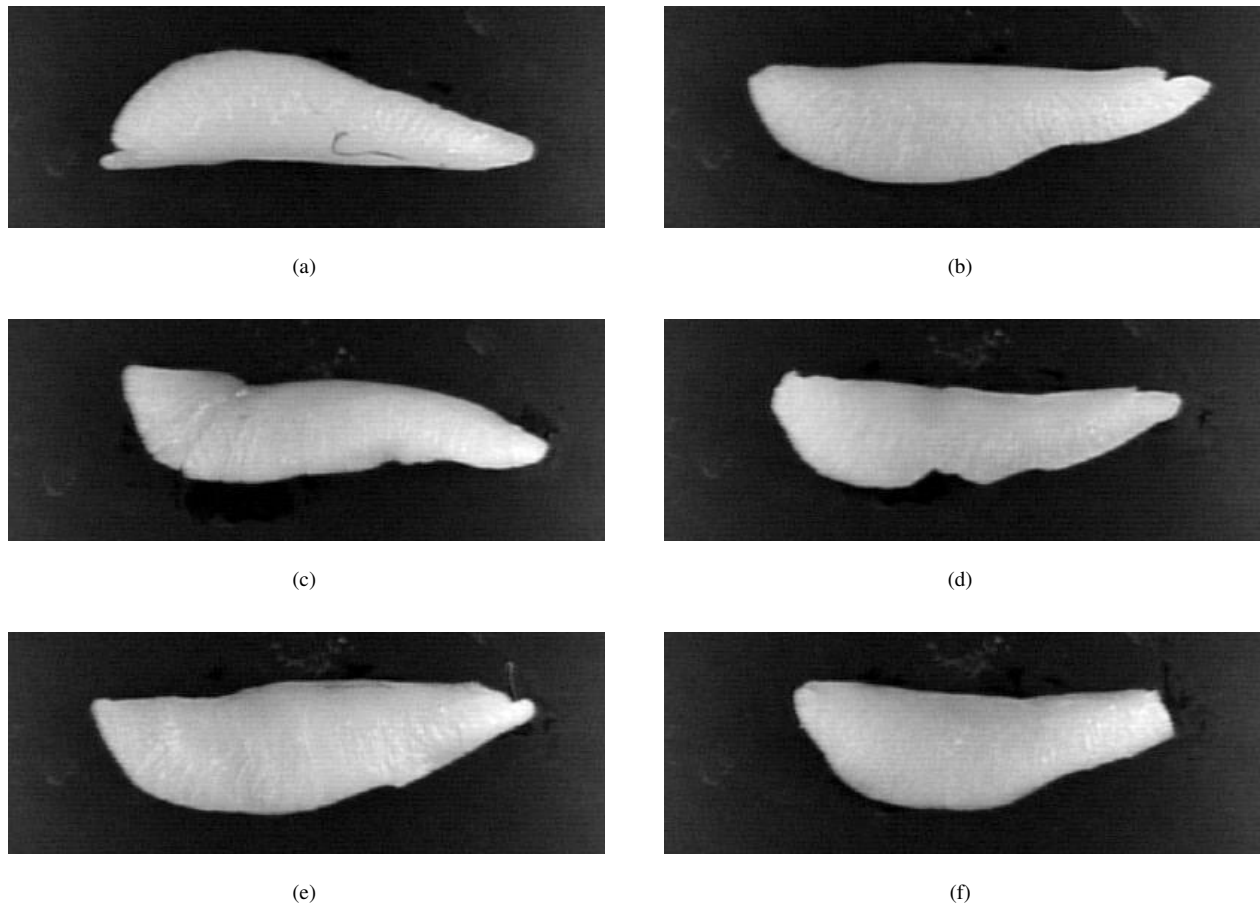


Fig. 7: Examples of herring roe classification grades imaged on-line under diffuse light conditions: (a) Grade 1; (b) Grade 1; (c) Grade 2-H, twist-cracks; (d) Grade 2-H, bites from belly; (e) Grade 2-H, flat, twist; and (f) Grade 2, broken tail.

— separating good roe from bad — and while Grade 1 roe are identified consistently, attempts to subclassify the defective roe have met with limited success. To address these limitations, additional information is required. Three dimensional and texture information would provide the system with features that are essential to proper classification.

The design of a second generation prototype, which can integrate the additional sensor information to better distinguish roe classifications, is the aim of a new effort. The requirements of the multisensor system for the grading of herring roe skeins are as follows:

1. Accurate determination of skein length (± 1 mm).
2. Estimation of skein weight ($5\text{--}50$ g ± 0.5 g).
3. Estimation of roe thickness as ratio of width.
4. Detection of parasite bites.
5. Detection of broken skeins (broken head or broken tail).

6. Detection of depressions ($> 12 \text{ mm}^2$).
7. Detection of twists ($> 12 \text{ mm}^2$).
8. Detection of proper yellow colour.
9. Detection of proper firmness of roe as an indicator of maturity.
10. Detection of bumps and curvature characteristics representative of cauliflower deformation.
11. Detection of cracks in the roe skin ($> 1 \text{ mm}$).

5.3. Object Model Development

From the requirements, there are eleven salient features that can be identified as necessary for classification. These include weight, length, thickness, firmness, presence/absence of parasite bite(s), breaks, cracks, twists, depressions, cauliflower, and proper colour. Many of these can be assessed on the basis of 2D visual information. Length, breaks, cracks, cauliflower bumps, parasite bites, and proper colour are all visible from an overhead view of the roe.

Thickness, twists, and depressions require information about the three dimensional profile of the roe skin. The 3D profile is usually represented as a surface map. Due to the variability of roe, thickness is represented as a ratio between the depth of the roe (as estimated by the 3D profile) and the width of the roe at the minor axis.

Weight cannot be measured directly. There are a number of systems which can measure mass as the product moves along the conveyor; however, they tend to suffer from two problems: one is the cost of such systems, and the other, more important, is inaccurate measurement of skeins with low weight ($< 10 \text{ g}$). As an alternative, the weight may be estimated using a linear regression model based on the peripheral length, area, and thickness of the roe [19].

Firmness is the only feature that does not lend itself to direct visual inspection. Traditionally, firmness has been assessed by handling the roe; however, this approach is not practical for an automated system [19]. Another method is the use of ultrasonic echo imaging. The strength of the echo signal is directly dependent on the structure, uniformity, and firmness of the object region which generates the echo. Therefore, the echo image contains features correlated with the firmness of the roe. These features may be extracted and used as an indirect measure.

None of the primary features are atomic. Each is broken down into the various atomic components which permit the detection of the feature. The primary features and corresponding subfeatures are shown in Figure 8.

The primary features are combined to produce eleven different classifications. Grade 1 roe is subclassified into six grades, according to weight. Each of these subclasses must be free of defect features, such as breaks and cracks. Additionally, Grade 1 roe must be firm, of the proper colour, and of sufficient size. If all of these criteria are satisfied, the roe is Grade 1; it is assigned to one of the subclassifications on the basis of weight.

For other grades, classification is dependent on the detection of certain distinguishing features. For example, the detection of a break will classify a piece of roe as Grade 2 provided the roe is also firm and of sufficient length.

5.4. Logical/Physical Sensor Selection

A logical sensor hierarchy is constructed from the object model starting with the primary features. Each of the features identified during the development of the object model is associated with an ELS which can extract the required feature. Because many of the primary features share common subfeatures, the logical sensor hierarchy, Figure 9, is considerably less complex than the object model. There are three physical sensors required: two CCD cameras and an ultrasonic probe; the details of each follow:

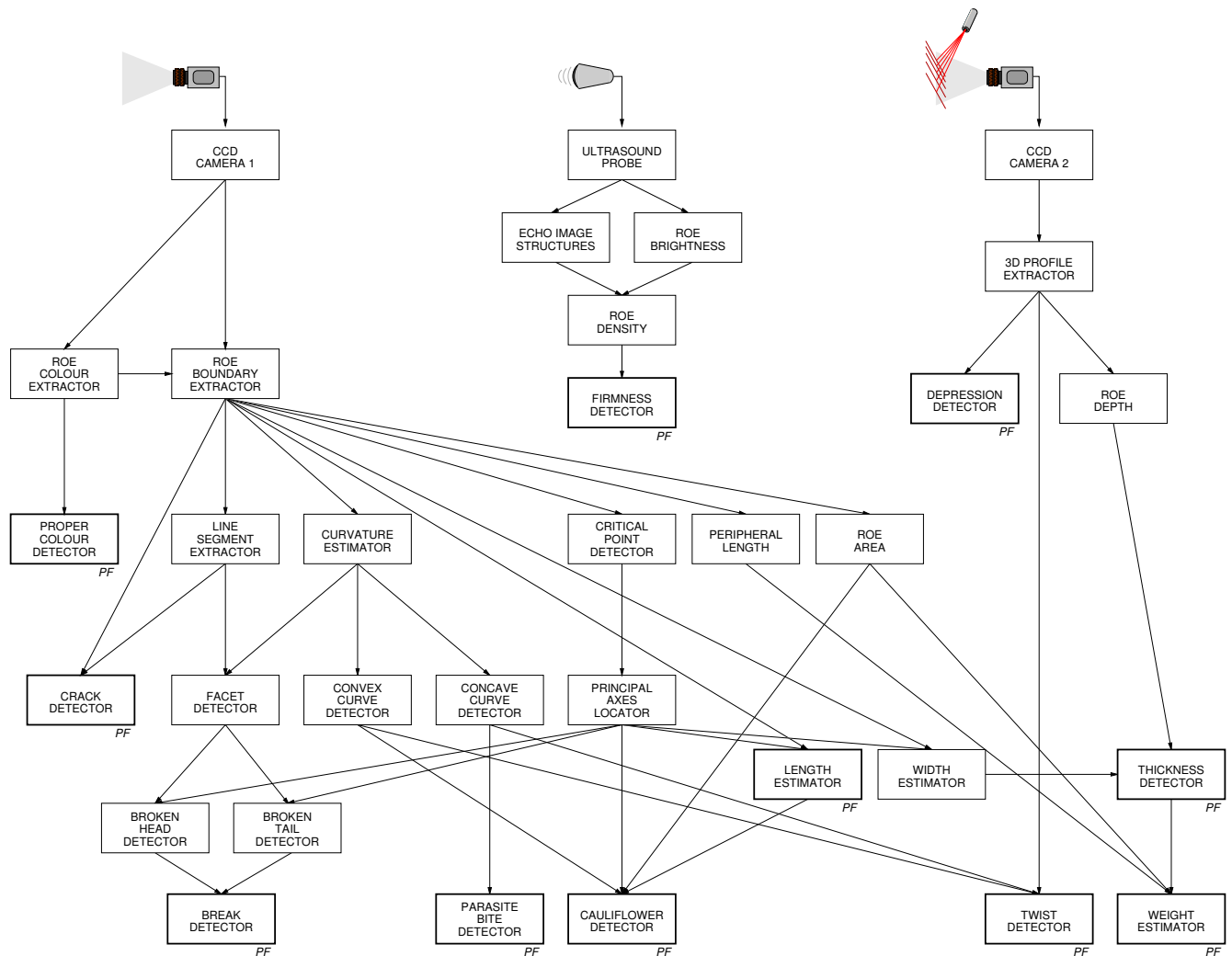


Fig. 9: Logical sensor hierarchy for herring roe grading. Sensors which provide primary features are outlined in bold and tagged *PF*.

The low-level subfeatures RGB Image and 2D Profile, from which a number of other subfeatures are derived, may be provided by a single colour camera with evenly distributed, diffuse lighting. Images obtained under such conditions are presented in Figure 7. A JVC TK1070U colour CCD camera with a 16mm f1:1.4 lens was used.

The Echo Image subfeature is provided by a 10 MHz mechanical sector ultrasound probe. This provides input to the ELSs

responsible for estimating of the firmness of a roe skein. Upon extracting the relevant image features, these are passed to the Firm ELS which uses fuzzy logic to estimate firmness.

The 3D Profile Extractor ELS utilizes a second CCD camera in combination with a structured laser light. Using knowledge of the image geometry, the 3D profile of the roe skein may be reconstructed.

5.5. Rulebase Definition

The rulebase generation follows from the object model. The object classifications outlined in Section 5.3. are used as the basis for the classification rules, Figure 10. The confidence membership function is used to express the confidence in the detection of the proper firmness, proper colour, breaks, cauliflower deformities, parasite bites, depressions, and cracks. The other features: length, weight, thickness, and twist, utilize specific membership functions.

```

IF Firm IS high AND Length IS normal AND ProperColour IS high AND Weight IS very very large AND ParasiteBite IS no AND Break IS no AND Depression IS no AND Twist IS no AND Crack IS no AND Cauliflower IS no AND Thickness IS normal THEN 3L-No1 = high
IF Firm IS high AND Length IS normal AND ProperColour IS high AND Weight IS very very large AND ParasiteBite IS no AND Break IS no AND Depression IS no AND Twist IS no AND Crack IS no AND Cauliflower IS low AND Thickness IS normal THEN 3L-No1 = high
IF Firm IS low OR ProperColour IS low OR ParasiteBite IS low OR Break IS low OR Depression IS low OR Twist IS medium OR Crack IS low OR Cauliflower IS low AND Weight IS very very large THEN 3L-No1 = low
IF Firm IS no OR ProperColour IS no OR ParasiteBite IS high OR Break IS high OR Depression IS high OR Twist IS high OR Crack IS high OR Cauliflower IS high AND Weight IS very very large THEN 3L-No1 = no
:
:
IF Firm IS high AND Length IS small AND Break IS high THEN No2 = high
IF Firm IS high AND Length IS normal AND Break IS high THEN No2 = high
IF Firm IS high AND Length IS small AND Break IS average THEN No2 = low
IF Firm IS high AND Length IS normal AND Break IS average THEN No2 = low
IF Break IS average THEN No2 = no

IF Firm IS high AND Length IS normal AND ParasiteBite IS high THEN No2-H = high
IF Firm IS average AND Length IS normal AND ParasiteBite IS high THEN No2-H = high
IF Firm IS high AND Length IS normal AND ParasiteBite IS average THEN No2-H = low
IF Firm IS high AND Length IS normal AND Depression IS high THEN No2-H = high
IF Firm IS average AND Length IS normal AND Depression IS high THEN No2-H = high
IF Firm IS high AND Length IS normal AND Depression IS average THEN No2-H = low
IF Firm IS high AND Length IS normal AND Twist IS high THEN No2-H = high
IF Firm IS average AND Length IS normal AND Twist IS high THEN No2-H = high
IF Firm IS high AND Length IS normal AND Twist IS medium THEN No2-H = high
IF Firm IS high AND Length IS normal AND Twist IS low THEN No2-H = no
IF ParasiteBite IS low AND Depression IS low AND Twist IS low THEN No2-H = no

IF Firm IS high AND Crack IS high THEN No2-C = high
IF Firm IS high AND Crack IS average THEN No2-C = low
IF Firm IS high AND Cauliflower IS high THEN No2-C = high
IF Firm IS high AND Cauliflower IS average THEN No2-C = low
IF Crack IS average AND Cauliflower IS average THEN No2-C = low
IF Crack IS low AND Cauliflower IS low THEN No2-C = no

IF Firm IS low THEN Unclassified = high
IF Firm IS average THEN Unclassified = low
IF Length IS very small THEN Unclassified = high
IF Length IS very very small THEN Unclassified = high
IF Length IS very small AND Weight IS small THEN Unclassified = high
IF Length IS very very small AND Weight IS very small THEN Unclassified = high
IF Length IS small AND Weight IS small THEN Unclassified = low

```

Fig. 10: Rules used to identify herring roe grades from primary features. For clarity, a number a rules used for classifying Grade 1 roe have been removed.

Once the roe have been classified, a decision is made about which bin it should be ejected into. The classifications are segregated using the same apparatus used by the prototype grading system. Bin 1 accepts Grade 1 Large, 2L, and 3L; Bin 2 accepts Grade 1 Medium; Bin 3 accepts Grade 1 Small; Bin 4 accepts Grade 2; Bin 5 accepts Grade 2-H; Bin 6 accepts Grade 2-C; and Bin 7 accepts all other grades and unclassified roe which fall off the end of the conveyor. Figure 11 presents the

rules which are used to infer this decision. The fuzzy membership functions associated with these rules are shown in Figure 12.

IF Grade1-3L IS <i>high</i> THEN Decision = <i>bin1</i>
IF Grade1-2L IS <i>high</i> THEN Decision = <i>bin1</i>
IF Grade1-Large IS <i>high</i> THEN Decision = <i>bin1</i>
IF Grade1-Medium IS <i>high</i> THEN Decision = <i>bin2</i>
IF Grade1-Small IS <i>high</i> THEN Decision = <i>bin3</i>
IF Grade1-Pencil IS <i>high</i> THEN Decision = <i>bin7</i>
IF Grade2 IS <i>high</i> THEN Decision = <i>bin4</i>
IF Grade2 IS <i>low</i> THEN Decision = <i>bin4</i>
IF Grade2-C IS <i>high</i> THEN Decision = <i>bin5</i>
IF Grade2-C IS <i>low</i> THEN Decision = <i>bin5</i>
IF Grade2-H IS <i>high</i> THEN Decision = <i>bin6</i>
IF Grade2-H IS <i>low</i> THEN Decision = <i>bin6</i>
IF Unclassified IS <i>high</i> THEN Decision = <i>bin7</i>
IF Unclassified IS <i>low</i> THEN Decision = <i>bin7</i>

Fig. 11: Rules used to determine decisions about how roe should be handled based on object classifications.

6. Conclusions and Future Work

ELSA is a multisensor integration architecture for industrial tasks. It is also, based upon the object model, a methodology for the construction of such a system. ELSA was developed to provide an organized approach for developing industrial-based sensor systems. It addresses the need for scalable, modular, and structured sensor systems, replacing current ad hoc approaches. The construction methodology enables domain experts, who lack signal processing knowledge, to design and understand a sensor system for their particular application.

To achieve this, ELSA is comprised of a number of different components. Extended Logical Sensors are presented as an improvement to the existing LS and ILS specifications. This improvement is realized by strongly encapsulating the ELS. The ELS may be polled by other sensors to determine its capabilities and request changes in the performance of the ELS, but its internal operation is hidden. Replacement sensors need only provide the same form of output. Other components, such as the Exception Handling Mechanism and the Integration Controller, serve to enhance the robustness and functionality of the architecture.

The object model used by ELSA is particularly suited to the representation of non-uniform products, or any object for which classification is desired. Objects are described in terms of their primary or distinguishing features. Primary features may be a composite of subfeatures. Objects are classified by using fuzzy membership functions to express how the primary features combine for each classification. The organization of the sensor system and the definition of the rulebase is driven by the object model.

Logical sensors are chosen to provide each of the features defined by the object model; this in turn determines what physical sensors are required by the system. The classification layer of the object model directly specifies how primary features are combined to determine object classifications. To demonstrate these concepts, ELSA was applied to the problem of herring roe grading. Development of a new prototype based on this architecture is ongoing.

The design and implementation of an ELS requires signal processing and programming knowledge that an industrial user

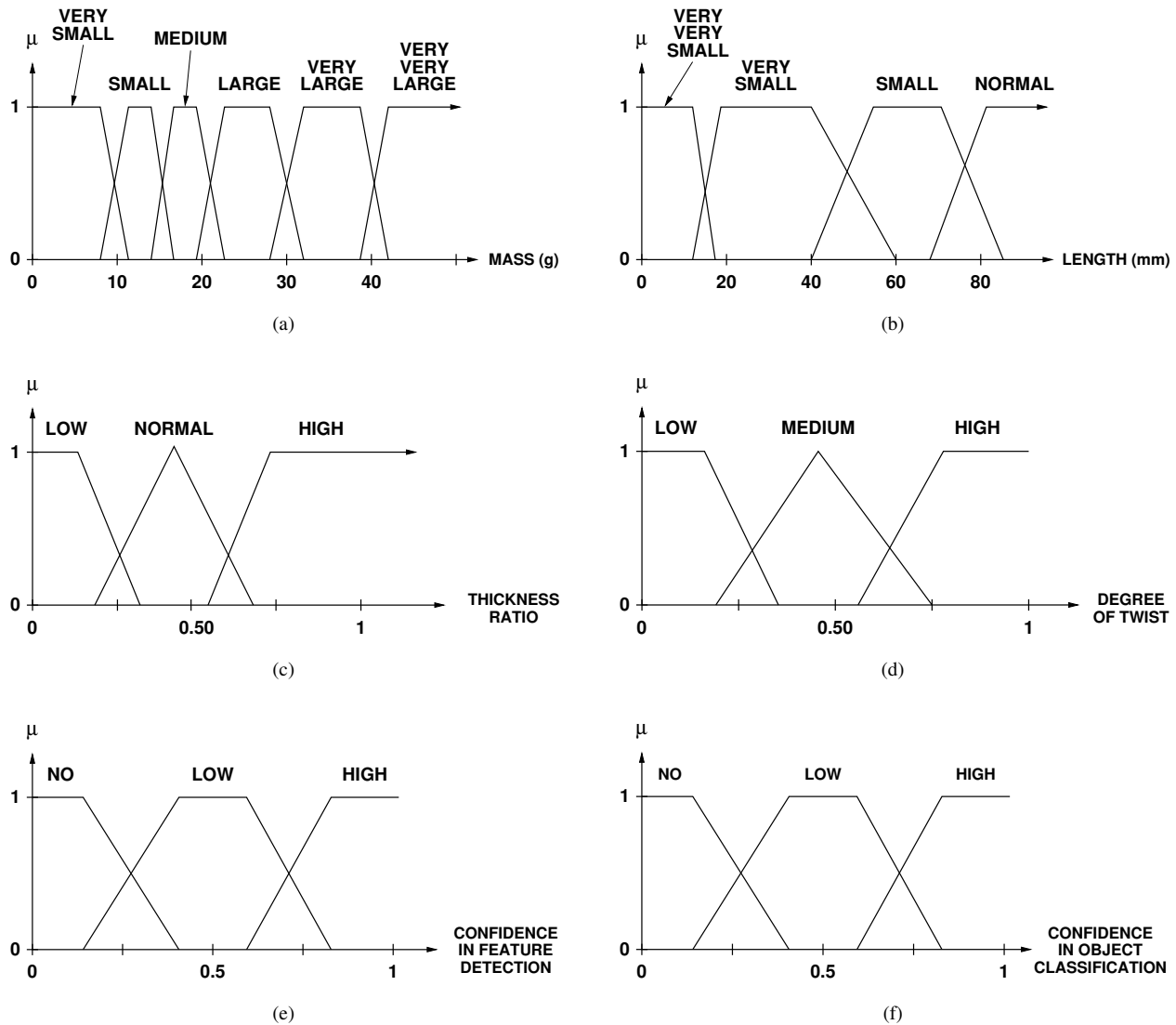


Fig. 12: Membership functions used for classification of herring roe grades: (a) weight; (b) length; (c) thickness; (d) degree of twist; (e) proper colour, firm, break, cauliflower, crack, depression, parasite bite; and (f) confidence in classification.

may not possess. Though this limits the ability of such a user to fully construct a system, it may be completely specified. This is because ELSA effectively separates the domain knowledge from the detailed sensor knowledge. If necessary, a technical expert may be consulted to develop the required ELS(s). Once a library of standard logical sensors is established for a set of applications, a system may be constructed without an in-depth understanding of the internal workings of each ELS. This makes ELSA particularly suitable for industrial users who wish to construct, modify, and maintain industrial multisensor systems.

Further work is required to extend ELSA to control applications. One approach to this may be the concept of a Logical Actuator (LA). Control decisions made by the Inference Engine would be passed to a LA hierarchy where directives are con-

verted into actions. The Logical Actuators thus serve as an interface between the high-level decision making system and the low-level process machinery. In this sense, a LA is an analogue to a LS. A similar idea, a combined Logical Sensor/Actuator (LSA) presented by Budenske and Gini [35]. By encapsulating the physical actuators, drivers, and planning algorithms, they may be altered without affecting the Inference Engine.

A library of Extended Logical Sensors will be constructed that is suitable for a variety of inspection and grading tasks. This will assist in the development of ELSA-based systems for applications such as the grading of herring roe, potatoes, blueberries, and other produce. Other applications include assembly, material handling, and machining operations.

Acknowledgements

The financial support of the Natural Sciences and Engineering Research Council of Canada and the Garfield Weston Foundation is gratefully acknowledged. The first author also gratefully acknowledges the support of the Gordon M. MacNabb Scholarship Foundation.

References

- [1] Beatty, D. A., 2D contour shape analysis for automated herring roe quality grading by computer vision Master's thesis, Department of Computer Science, University of British Columbia, Vancouver, B.C. V6T 1Z4, (1993).
- [2] Davies, E. R., *Machine Vision: Theory, Algorithms, Practicalities*, 2nd edn. Academic Press, San Diego, CA (1997).
- [3] Luo, R. C. and Kay, M. G., Data fusion and sensor integration: State-of-the-art 1990s. In *Data Fusion in Robotics and Machine Intelligence* (Edited by Abidi, M. A. and Gonzalez, R. C.), pp. 7–135. Academic Press, San Diego, CA (1992).
- [4] Murphy, R. R. and Arkin, R. C., SFX: an architecture for action-oriented sensor fusion. In *Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems* **2**, 1079–1086 (1992).
- [5] Murphy, R. R., Biological and cognitive foundations of intelligent sensor fusion. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans* **26**, 42–51 (1996).
- [6] Bower, T. G. R., The evolution of sensory systems. In *Perception: Essays in Honor of James J. Gibson* (Edited by MacLeod, R. B. and Pick Jr., H. L.), pp. 141–153. Cornell University Press, Ithaca, NY (1974).
- [7] Lee, S., Sensor fusion and planning with perception-action network. In *Proceedings of the 1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 687–696 (1996).
- [8] Iyengar, S. S., Prasad, L., and Min, M., *Advances in Distributed Sensor Technology*. Prentice Hall PTR, Upper Saddle River, NJ (1995).
- [9] Queeney, T. and Woods, E., A generic architecture for real-time multisensor fusion tracking algorithm development and evaluation. In *Proceedings of the SPIE - Signal Processing, Sensor Fusion, and Target Recognition VII* **2355**, 33–42 (1994).
- [10] Henderson, T. C. and Shilcrat, E., Logical sensor systems. *Journal of Robotic Systems* **1**, 169–193 (1984).
- [11] Henderson, T. C., Hansen, C., and Bhanu, B., The specification of distributed sensing and control. *Journal of Robotic Systems* **2**, 387–396 (1985).
- [12] Weller, G. A., Groen, F. C. A., and Hertzberger, L. O., A sensor processing model incorporating error detection and recovery. In *Traditional and Non-Traditional Robotic Sensors* (Edited by Henderson, T. C.), **F 63**, pp. 351–363. Springer-Verlag, Berlin (1990).

- [13] Groen, F., Antonissen, P., and Weller, G., Model based robot vision by extending the logical sensor concept. In *1993 IEEE Instrumentation and Measurement Technology Conference*, 584–588 (1993).
- [14] Dekhil, M. and Henderson, T. C., Instrumented sensor system architecture. *The International Journal of Robotics Research* **17**, 402–417 (1998).
- [15] Gunasekaran, S., Computer vision technology for food quality assurance. *Trends in Food Science & Technology* **7**, 245–256 (1996).
- [16] Daley, W., Carey, R., and Thompson, C., Poultry grading/inspection using color imaging. In *Proceedings of the SPIE - Machine Vision Applications in Industrial Inspection* **1907**, 124–132 (1993).
- [17] Calpe, J., Pla, F., Monfort, J., Díaz, P., and Boada, J. C., Robust low-cost vision system for fruit grading. In *Proceedings of the 1996 8th Mediterranean Electrotechnical Conference* **3**, 1710–1713 (1996).
- [18] Cao, L. X., de Silva, C. W., and Gosine, R. G., A knowledge-based fuzzy classification system for herring roe grading. In *Proceedings of the Winter Annual Meeting on Intelligent Control Systems DSC-48*, 47–56 (1993).
- [19] Croft, E. A., de Silva, C. W., and Kurnianto, S., Sensor technology integration in an intelligent machine for herring roe grading. *IEEE/ASME Transactions on Mechatronics* **1**, 204–215 (1996).
- [20] Heinemann, P. H., Pathare, N. P., and Morrow, C. T., An automated inspection station for machine-vision grading of potatoes. *Machine Vision and Applications* **9**, 14–19 (1996).
- [21] Luzuriaga, D. A., Balaban, M. O., and Yeralan, S., Analysis of visual quality attributes of white shrimp by machine vision. *Journal of Food Science* **62**, 113–118 (1997).
- [22] Brown, G., Forte, P., Malyan, R., and Barnwell, P., Object oriented recognition for automatic inspection. In *Proceedings of the SPIE - Machine Vision Applications in Industrial Inspection II* **2183**, 68–80 (1994).
- [23] O’Dor, M. A., Identification of salmon can-filling defects using machine vision Master’s thesis, Department of Mechanical Engineering, University of British Columbia, Vancouver, B.C. V6T 1Z4, (1998).
- [24] Dipix Technologies Inc., QualiVision product information (1998). <http://www.dipix.com/vissys/vissset.htm>.
- [25] Flexible systems for trimming and portioning. *World Fishing* 13–14 (1997).
- [26] Key Technology, Inc., Product catalog (1995).
- [27] Naish, M. D. and Croft, E. A., Data representation and organization for an industrial multisensor integration architecture. In *Proceedings of the 1997 IEEE International Conference on Systems, Man, and Cybernetics* **1**, 821–826 (1997).
- [28] Chavez, G. T. and Murphy, R. R., Exception handling for sensor fusion. In *Proceedings of the SPIE - Signal Processing, Sensor Fusion, and Target Recognition VI* **2059**, 142–153 (1993).
- [29] de Silva, C. W., *Intelligent Control: Fuzzy Logic Applications*. CRC Press, Boca Raton, Florida (1995).
- [30] Lippmann, R. P., An introduction to computing with neural nets. *IEEE ASSP Magazine* 4–22 (1987).
- [31] Biederman, I., Recognition-by-components: A theory of human image understanding. *Psychological Review* **94**, 115–147 (1987).
- [32] Tomita, F. and Tsuji, S., *Computer Analysis of Visual Textures*. Kluwer Academic Publishers, Norwell, Massachusetts (1990).
- [33] Zadeh, L. A., Fuzzy logic = computing with words. *IEEE Transactions on Fuzzy Systems* **4**, 103–111 (1996).
- [34] Shigley, J. E. and Mischke, C. R., *Mechanical Engineering Design*, 5th edn. McGraw-Hill, New York (1989).
- [35] Budenske, J. and Gini, M., Sensor explication: Knowledge-based robotic plan execution through logical objects. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics* **27**, 611–625 (1997).