# An Architecture for Sensor Data Fusion to Reduce Data Transmission Bandwidth

**Dale Lord and Kurt Kosbar**

**Telemetry Learning Center**
**Department of Electrical and Computer Engineering**
**University of Missouri – Rolla**
**Rolla, MO 65409-0040**

## ABSTRACT

Sensor networks can demand large amounts of bandwidth if the raw sensor data is transferred to a central location. Feature recognition and sensor fusion algorithms can reduce this bandwidth. Unfortunately the designers of the system, having not yet seen the data which will be collected, may not know which algorithms should be used at the time the system is first installed. This paper describes a flexible architecture which allows the deployment of data reduction algorithms throughout the network while the system is in service. The network of sensors approach not only allows for signal processing to be pushed closer to the sensor, but helps accommodate extensions to the system in a very efficient and structured manner.

Keywords: Sensor Networks, Data Fusion, Feature Extraction, Network Architecture

## INTRODUCTION

Systems are often designed with hundreds, if not thousands, of sensors. Theses sensors can monitor anything from the occasional event switch which may produce data once a month, to video images and other high bandwidth sensors that produce megabits of information per second. It can be a challenge to decide how to process, forward, display and archive this data.

Remote monitoring engineers often require all the data as rapidly as it can be sampled in order to perform initial system monitoring. This leads to high communications costs with any transmission medium, and is especially difficult for wireless applications. Having rapid access to the raw data may be necessary during early system design and verification, so abnormal operation can be quickly detected and analyzed. Often, as remote monitoring engineers determine which sensors and what algorithms are the best indicators of system anomalies, transmitting most of the data becomes unnecessary. To reduce the overall bandwidth it is

desirable to move these algorithms to the remote sensors so that only the result of this detection algorithm needs to be transmitted.

The goal of this research is to design a system that allows online improvements in the monitoring of sensor data. Factors considered include a simple extendable architecture, transformation reusability to reduce overall system requirements, and adaptability to include a wide variety of sensors.

Starting at the sensor data, each sensor is sampled and controlled by a sensor server process. These servers are written in such a way that they can provide data to multiple client processes with potentially different sample rates. These client processes are built to transform the sensor data into useful information.

Once the data is transformed into information that requires action a final level of processes have control over the communications channels to the local as well as remote users. These processes can send alerts as well as data to the users so that impending failures or events can be managed. In some cases the system could even have the ability to shut down operations in order to save money and lives.

## SYSTEM ARCHITECTURE

The system architecture is described as separate processes which may be distributed among various processors. Most of the systems are described how they would act on a typical embedded computer architecture with system random-access memory (RAM), hard drive and various input/output (IO) devices. However, many of the processes can be implemented on a microcontroller system with RAM, flash memory storage and the various IO.

Sensor server processes are used to collect various input data and trigger sensor processes to transform and analyze the data. It is important to note that these sensor processes are kept separate from the server process for a reason. Often the system may have several tasks to exhibit, and multiple goals to achieve. However, these tasks and goals may need the same transform on the sensor data. By keeping the transform process separate, the transform is only performed once. Also, more than one sensor-process may need access to the same sensor data to analyze different aspects of the raw sensory data. This type of data flow and processing is the beginning of a web of processes and tasks that collect refine and analyze data.

Each output is controlled by a process that has the responsibility to schedule different task communications based upon their priority. A final task coordinator is used to add new processes to the active system. New processes register with the task coordinator which in turn adds the triggers to the existing processes to start the new process.

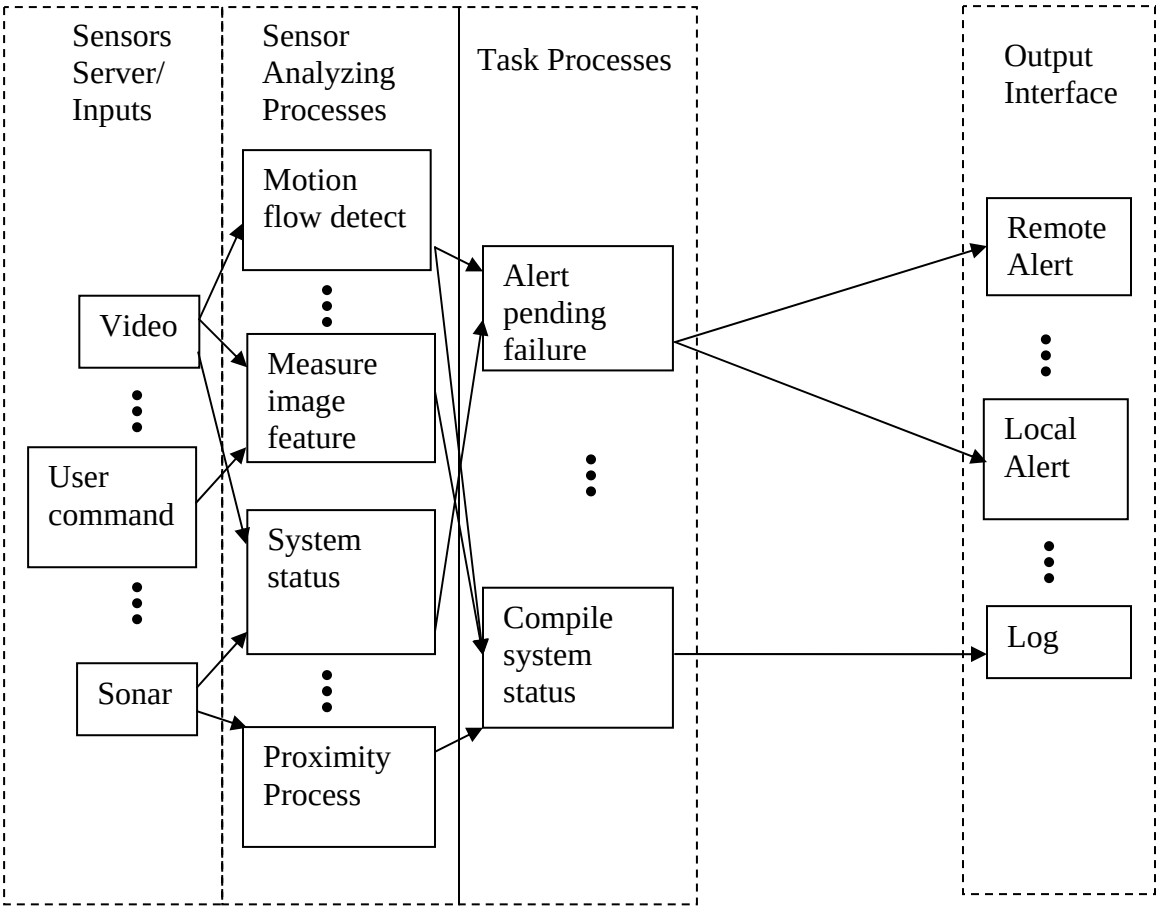


Figure 1 System architecture

A schematic of the proposed systemic system architecture is shown in Figure 1. Each of the sensors is controlled by sensor server and provides raw data which is analyzed by sensor processes to extract useful information. Note that some raw data from a given sensor may be processed by multiple sensor processes in order to extract different information. For example video data can be processed by a motion flow process to detect moving objects, while the same raw data is being processed by texture segmentation process to detect the extent an object. Note that the same sensory information can be used by multiple tasks and each task may also act using information from multiple sensors.

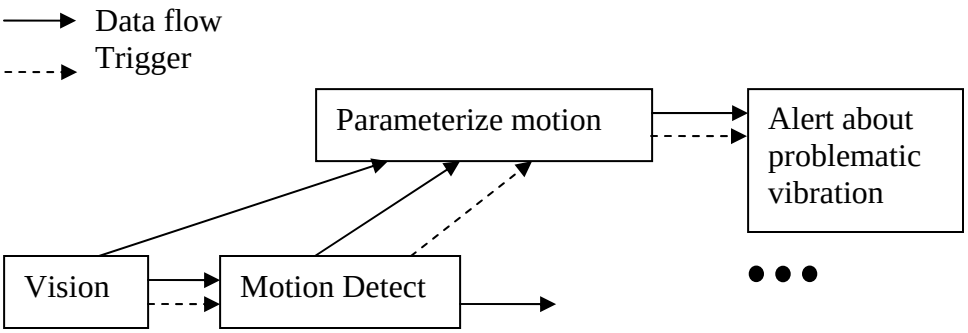


Figure 2 Example system flow

Figure 2 is a simple example set of processes that can be used to detect problematic vibration in a system. The system flow starts at the camera where a sensor server obtains images and makes them available to image processing tasks. Each new image triggers the image processing tasks to start processing on the new image. In this example a motion detection process takes successive images and looks for motion. Once a threshold is exceeded a trigger is sent to activate any dependent processes. The data is made available to these new processes so that further analysis can be done. In this example the system could determine the location of the vibrating parts and calculate the maximum acceleration in the vibration. This would then be sent to an output channel for alerting the remote user.

## SENSOR SERVERS

Many times there are multiple tasks that could require sensor information. For example; the system may be using the vision information to estimate an objects position while at the same time it is required to visually measure an objects size. Without a vision server allowing for concurrent connections to the vision data, these two tasks would have to be integrated into a single process or take turns using the video resource. This may not be bad for two tasks, but in general there could be dozens of visual tasks that happening at the same time and each task would require different amounts of processing time with different data. This multiple access problem is solved by the use of sensor servers.

One common sensor is a camera, so the implementation of a sensor server is described in detail using the data obtained by a camera. Allowing multiple processes to access the same image information proposes some problems. Some processes might take longer than a single frame time period of 33ms to work with the image. If the image capturing were delayed until all processes were done with the image data, then the capture rate would not be fast enough for some high speed sensor processes like motion detection. While making a copy of the image is possible, copying takes a significant amount of processor time and would be required for each process that needed the image data. The solution is to implement a ring buffer. The video capture is done independent of the image data usage so the frame rate is always maintained at thirty frames per second.

This is done is by filling the ring buffer with frames using the capture card's Direct Memory Access controller. Once the buffer is full, normal operation commences. The server blocks (waits) until the next to the last frame is done transferring from the capture card. The capture card continues with the last frame however before it is done the next frame for it to transfer is queued to replace the next oldest frame in the ring buffer. By this method the frames are available for the maximum amount of time and the DMA controller always has the next frame to transfer before it is done with its current frame (Figure 3).

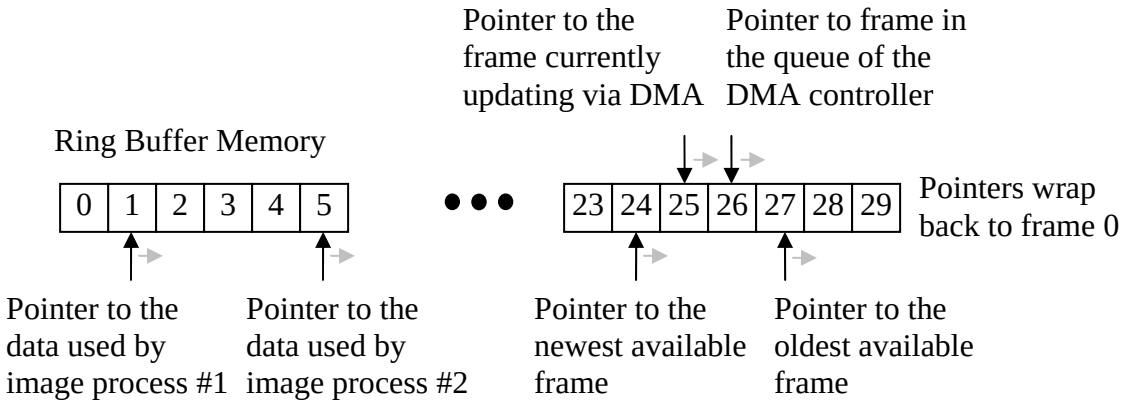


Figure 3 Video ring buffer operation

The size of the ring buffer multiplied by the time per frame gives the maximum time for processing a specific image. Typically thirty frames are held so that there is a complete second of processor time to analyze the image data.

Other sensor processes have similar requirements of being able to access the data for more time than the update rate. And even if processing the data from a sensor may not take longer than its update rate, often the short term history of sensor data is needed. For example; after some image processing, a task might reveal a situation where other data is needed. At that point the other data needs to have a ring buffer that maintains data at the same time as the image data.

For that reason, the other sensors of the system need to have a ring buffer with a size that allows their data to be available for a similar amount of time. These sensor servers are considered an extension of the sensor hardware and are most useful if they are placed on processors at or in close proximity to the sensors.

## SENSOR PROCESSES

Some sensors give useful information directly from their outputs. For example sonar proximity sensors use the time that the sound takes to travel away from and bounce back to a transducer. This time directly relates to a distance to an obstacle. Cameras on the other hand are primarily used to reproduce a scene on a screen and therefore there is little knowledge gained from the raw data of an image. Processing is required to gather edges, segment areas and recognize features.

This process of going from raw data to usable knowledge is highly dependent on the sensor and the knowledge that is desired. It is important to note that the image sensor processing done in these algorithms is not typically done on the whole image of each frame. Most often there is apriori knowledge of what is expected in a view. For example in processing could concentrate on the area in the image where motion is detected.

## TASK PROCESSES

Task process use information from various sensor processes to draw a conclusion about the sensor data and trigger some form of output. From the remote monitoring stand point this would be the collection of all the indications that would indicate an impending failure.

## OUTPUT INTERFACE

With the system having various tasks and tasks occurring simultaneously the system must manage its outputs. For example if one task is communicating the system status and a second task needs to alert that there is an impending failure the output interface will abort the status communication and allow the higher priority message through.

## TASK COORDINATOR

The task coordinator also has the job of registering tasks and its triggers. When a task is added to the system there are one or more processes than can trigger the new task and there may be one or more other tasks that need to be triggered after the new task. For example if a task is added to calculate the average system pressure during startup, a trigger would be added to the startup sensor server to start the calculation of average system pressure. Each new task and its associated triggers can be added and removed from the system after deployment.

## CONCLUSIONS

In situations where there is a cost associated with the total amount of data transmitted or a large cost for a high bandwidth channel, it is desirable to minimize the data sent through this communications channel. It is often possible to reduce data to summaries of data or to use feature detection to detect and alert when problems occur. In the proposed architecture a system of adding processes that prepare and process sensor data in a distributed manner not only allows for the system to be extended as new anomaly detection algorithms are developed it also allows for parts of the processing to be pushed close to the sensor and reducing the overall communications requirements.

## REFERENCES


Antony, Richard T. “Principles of Data Fusion Automation” Artech House Inc. 1995 ISBN 0-89006-760-0

Choset, H., Konukseven, I. and Rizzi, A. “Sensor Based Planning: A Control Law for Generating the Generalized Voronoi Graph,” Proc. Eighth IEEE Int’l Conf. Advanced Robotics, pp 333-8, 1997

Chown, E., Kaplan, S. and Kortenkamp, D. “Prototypes, Location, and Assocciative Networks (PLAN): Towards a Unified Theory of Cognitive Mapping,” Cognitive Science, vol 19, no. 1, pp. 1-51, January 1995

Hall, David L. “Mathematical Techniques in Multisensor Data Fusion,” Arrech House Inc. 1992 ISBN 0-89006-558-6

Horn, Paul, Klaus, Berthod “Robot Vision,” The MIT Press McGraw-Hill Book Company 1997 ISBN 0-262-08159-8

Irving, William W. and Tsitsiklis, John N “Some Properties of Optimal Thresholds in Decentralized Detection,” IEEE Transactions on Automatic Control vol.39 no.4 April 1994

Jack, Keith “Video Demystified” High Text Interactive 1996 ISBN 1-878707-X

Luger, George F and Stubblefield, William A “Artificial Intelligence – Structures and Strategies for Complex Problem Solving,” Addison-Wesley 1998 ISBN 0-805-31196-3

Manian, Vidya, Vasquez, Ramon and Katiyar, Praveen “Texture Classification Using Logical Operators,” IEEE Transactions on Image Processing vol. 9 no. 10 October 2000

Mirmehdi, Majid and Petrou, Maria “Segmentation of Color Textures,” IEEE Transactions on Pattern Analysis and Machine Intelligence vol. 22 no. 2 February 2000

Moravec, H.P. and Elfes, A. “High Resolution Maps form Wide Angle Sonar,” Proc. IEEE Int’l Conf. Robotics and Automation, pp. 116-121 September 1995

MySQL (viewed 11-21-2002)
http://www.mysql.com

Paschos, George “Perceptually Uniform Color Spaces for Color Texture Analysis: An Empirical Evaluation,” IEEE Transaction on Image Processing vol. 10, no. 6 June 2001

Poloriod 6500 Series Sonar Ranging Module PID #615077

“Programming in C UNIX System Calls and Subroutines,” (viewed 11-6-2002) http://www.cs.cf.ac.uk/Dave/C/CE.html

Shafer, S., Stentz, A., and Thorpe, C. "An Architecture for Sensor Fusion in a Mobile Robot," tech. report CMU-RI-TR-86-09, Robotics Institute, Carnegie Mellon University, April, 1986.

Sony "EVI-30 camera Brochure," number 99B-a

Sony "EVI-30 camera Operations Manual," number 3-856-663-04

Uchiyama, Toshio, Mukawana, Naoki and Kaneko, Hiroshi "Estimation of Homogeneous Regions of Segmentation of Textured Images," IEEE Transactions on Pattern analysis and Machine Intelligence 2000

"The Video for Linux Two Video Capture API Specification" (viewed 11-6-2002) http://www.thedirks.org/v4l2/v4l2cap.htm