Proceedings of ICAR 2003
The 11th International Conference on Advanced Robotics
Coimbra, Portugal, June 30 - July 3, 2003

Fusion of Lidar and Stereo Range for Mobile Robots

Kevin M. Nickels
Department of Engineering Science
Trinity University
San Antonio, TX, USA

Andrés Castaño and Christopher Cianci
NASA/Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA, USA

Abstract

This project involves the use of two physical sensors and several modular algorithms to generate a single spatial representation of obstacles and free-space surrounding a mobile robot. This representation is used to track obstacles over time.

We define a unified architecture that utilized a spatial object representation as the default communication conduit between all modules. This representation includes an occupancy grid of the area immediately surrounding the robot, along with geometric scaling and density information, and confidence estimates for these data. The flexibility of this architecture has been demonstrated by utilizing both live and recorded data sets from different lidar sensors, cameras, and processing modules, with minimal or no changes to the processing code.

This system shows promise as a flexible sensor fusion framework for mobile robotics. Advantages to this system over existing systems include the ability to encapsulate sensors, so that downstream algorithms don't need to know the details of the sensor suite on the robot, and for the system to automatically adjust for conditions where a single sensor fails or performs poorly but other sensors are functioning properly, such as in open fields where the range of lidar data is quite limited or in grassy areas where stereo range has well-known problems.

1 Introduction

It is often useful in mobile robotics to fuse data from multiple, possibly heterogeneous, sensors. In this work, we consider an architecture targeted at tracking traversibility and obstacle information for the purposes of navigation. This architecture is intended to be extensible and to simplify the change or addition of physical sensors as well as additional processing modules. The framework utilizes data encapsulation/data hiding from physical sensors, as described in Section 2, aiding prototyping and scenario playback. This also aids the development of modules that process obstacle information without needing information about the genesis of that information.

A fusion of sensor data can happen at the data, the feature, or the decision level [4]. Data fusion occurs when the raw data from various sensors are combined without significant post-processing. Feature fusion occurs when features are extracted from data at the sensor level before combinination. Fusion at the decision level requires that the fuser algorithm accumulate data from all sensors before fusing. Our algorithm fuses at the feature and data levels, depending on the sophistication of each physical sensor module.

Existing algorithms used include JPL-Stereo, a range-from-stereo package [1] and foliage-detector, a foliage filter for lidar data [3], both developed by the Machine Vision group at JPL, and several supporting libraries. The system was developed with flexibility and extensibility in mind, and this was demonstrated by the addition and integration of the stereo sensor suite to the lidar/foliage framework with an integration time of approximately two days.

A new algorithm, described in Section 2.4, was developed to intelligently fuse the range data from multiple modules: stereo range, lidar range, lidar range filtered by foliage detection, an object tracking module, and the previous fused summary map. An object tracker was developed based on optimal estimation theory and modern radar tracking systems [2] that tracked obstacles over time in this summary map.

2 Sensor Abstraction

Each processing block in our algorithm is defined to be a *sensor*. Sensors can be physical, as in the case of a lidar range finder, or virtual, as in the case of a foliage detector module or the traversibility estimation mentioned above. Each sensor takes as input a map or maps from another sensor and/or an input stream from a physical sensor. Each sensor provides (upon request) one or several maps as output. Each sensor must provide a minimal set of methods, including:

- `trigger()` - process the next set of data
- `get_capabilities()` - return a string describing the sensor
- `get_map(n)` - return the n^{th} map

```
map (type T):

 T data(*,*);    // sensor-defined
 uchar conf(*,*);// confidence
 timeval ts;     // data timestamp
 int dx,dy;      // cell size (mm)
 int sizex,sizey; // mapsize (cells)
```

Figure 1: *Definition of Map*

Each sensor may provide additional methods as deemed appropriate. Each sensor provides access to one or several output maps. These maps may illustrate, for example, the type of vegetation found or the reflectance returns from a lidar scan for each cell in the grid. These additional data, encapsulated in a map, may be utilized by this sensor or any other sensor to generate additional maps as described below.

2.1 Maps as Data Fusion Elements

The vehicle proposed in this work for the fusion of this disparate information is the *sensor map*, as defined in Figure 1. This data structure is a spatial environment model, with each data element representing some area in the world, some data about that region of the world, and a confidence measure of this classification. Each map has a minimum requirement to estimate the (binary) occupancy status of a cell (this map is commonly referred to as an occupancy grid [7]), and a confidence of that status. Each sensor also has the ability to define additional data on its sensor map, different maps, or even additional non-map information (such as the vector of angle/range pairs, in the lidar sensor). For example, traversibility might be estimated for each region based on the local texture found in a visual image of that region, as done in [1], and the traversibility tag encoded as a map.

2.2 History Mechanism

Each sensor can request that any other sensor keep track of old maps for the purposes of temporal filtering. Each sensor keeps a matrix of history requests, including its own history preferences (for example, the foliage filtering scheme described in [3] utilizes the past two scans as well as the current scan). This enables the system to keep only as many past maps as necessary by eliminating multiple copies of maps.

2.3 Foliage Detection Sensor

As an example of a virtual sensor, the definition of the foliage_detection sensor is given in Figure 2. The non_foliage map contains filtered versions of the lidar return data, where some heuristics have been used to throw out lidar range returns that correspond to grass. A detailed description of this algorithm is given in [3], and an alternative statistical formulation is given in [8].

```
sensor foliage_detection:

inputs:
  map raw_lidar (type boolean);
outputs:
  boolean alarm;
  map non_foliage (type boolean);
methods:
  trigger();
  data_ready();
  get_capabilities();
  get_map (n);
configuration:
  set_distances(dmin,dmax);
```

Figure 2: *Definition of Foliage Detection Sensor*

2.4 Summarizer Sensor

There is a virtual sensor called the summarizer that agglomerates the information from all of other sensors (including the previous summary map) to create a summary map. This summary map represents the aggregate best information available from all the sensors that are on-line and operating correctly. Each physical sensor module is responsible for providing safety checks so that if the hardware malfunctions or is not operational, the confidence values associated with its data are set appropriately. The summarizer combines the information (and associated confidence levels) from the sensors with the summary map from the previous time step to obtain a system-level estimate of the map occupancy.

The summarizer has a rule base from which to determine the appropriate weighting of sensor data, in addition to the self-expressed confidence of these data. For example, if the foliage_detection sensor is on-line, the impact of the raw_lidar data is considerably reduced. The weighting of the previous summarizer data relative to the other sensors determines the amount of *memory* the system has. This can be adjusted according to *a-priori* assumptions about the environment of the robot, and the data frame rate of the sensor suite. In this way reasonable summary maps can be generated no matter what combination of sensors is operational at a given point in time.

In particular, the summary map data for cell (i, j) is computed as:

$$d(i,j) = \sum_{f}[mw_s \times c_s(i,j)] > \sum_{e}[mw_s \times c_s(i,j)] \quad (1)$$

$$c(i,j) = \sum_{w} c_s(i,j) / \sum_{w} 1 \quad (2)$$

where $0 \leq mw_s \leq 1$ indicates the map weighting described above, $d(i,j)$ is a binary-valued entity (1=full,0=empty), $c_s(i,j)$ is the sensor-expressed data self-confidence, and the summary indices are as follows:

- f: $s \ni d_s(i,j)$ =full (sensor set voting *full*)
- e: $s \ni d_s(i,j)$ =empty (sensor set voting *empty*)
- w: $s \ni d_s(i,j) = d(i,j)$ (sensor set agreeing with winning vote.)

The index (i,j) is adjusted in the case that the different maps being summarized have differing extents and/or densities. The summary map therefore is of the maximal extent of any of the component sensors, and of the maximal density.

This arrangement has several potential benefits over an explicit and direct estimation of map occupancy as done, e.g., in [5]. The summary map structure is robust to sensor dropout; to similar but non-identical sensor replacement, such as the replacement of a failed component in the field; to degraded performance of a single sensor, for example due to occlusion; and to changing sensor suites. The summarizer algorithm makes use of whatever sensors are available and operating, without requiring recoding.

2.5 Object Tracking

An object tracker has been developed that utilizes the summary map for input. This tracker defines a *track scoring function* of

$$\begin{aligned} TS(k) = TS(k-1) \\ + ln(P_d V_c/(P_{fa}\sqrt{|\mathbf{S}|})) \\ - ln(2\pi) + d^2/2 \end{aligned} \quad (3)$$

where $d = \tilde{\mathbf{y}}'\mathbf{S}\tilde{\mathbf{y}}$ is the normalized distance between the observation and the prediction, P_d, P_{fa} are the a-priori probabilities of detection and false alarms, and V_c is the volume of the tracking space. This is a standard scoring function for an detection-only sensor [2]. Many standard definitions exist for this *distance* metric. For point targets, euclidean or range-azimuth distance could be used. Blackman and Popoli discuss several distance metrics and their implications in [2]. This function serves to decrease the score according to the expected temporal degradation of the estimate over a time step, and to increase it according to the quality of the new measurements. In addition, we define a track confirmation threshold and a track deletion function that are roughly equivalent to a M/N rule of $4/5$[1] and a T_D deletion threshold of 5[2]. Again, this is a standard formulation for modern radar tracking systems.

At each step, we generate a set of returns by thresholding the confidences on the occupied cells of the summary map. These returns are clustered according to some spatial heuristics. The resulting observations are used to create an observation to track (OTT) assignment matrix, as described in [2]. The OTT is an $N_T \times N_T + N_O$ matrix, where N_T is the number of existing tracks and N_O is the number of observations. The first N_T rows of the matrix associate each observation with each track. If observation O_i satisfies an ellipsoidal data gate generated from track T_j, $OTT(i,j)$ is set to the margin by which the gate was satisfied. Otherwise, $OTT(i,j)$ is set to -1. The final N_O rows of the OTT consider the creation of new tracks for each observation. For these rows, $OTT(i, N_T+i)$ is set to 0 and the remaining entries to -1. Finally, each column of the OTT is considered to determine the optimal track assignment for the corresponding observation (i.e. the highest OTT entry in that column — for an existing or new track). Any track without an observation assigned gets a track score reduction.

Finally, track maintenance is performed. New tracks are added to the active track list. Tracks which satisfy the track deletion function are removed. Tracks with scores above the track confirmation threshold are labeled as confirmed. All confirmed tracks are mapped on the tracker output map, and processing ceases until the next set of sensor observations.

The object tracker is not particularly sophisticated at this time, the main purpose being to provide robustness to target dropout. The foliage filtering mechanism and range-from-stereo sensors are both fairly conservative in their detection of obstacles. This leads to *target dropout*, where a target will be missed on a few frames between several strong returns. The object tracker will presume persistence of the target at constant velocity, while decreasing the track score with each time step without a paired observation, until the target reappears or the track deletion function takes effect. The foliage detector in particular has a large occurrence of false alarms (i.e. foliage returns that are classified as non-foliage) in a given frame, but the spatial and temporal distribution of these returns is random. This leads to a large number of tracks that are created but never confirmed. These tracks are deleted after a small number of time steps.

The object tracker sensor is useful to track temporally and spatially persistent objects in the summary map. This is true whether the summary map is comprised of data from a single lidar sensor or from the entire suite of sensors and signal processing algorithms described above. Indeed, this is the intent

[1] The M/N track confirmation/deletion rule states that a track is confirmed after M successive observations in the last N steps.

[2] The T_D deletion threshold states that a track is deleted after T_D consecutive time steps without a paired observation.

of the summary map and the framework surrounding it: downstream processes need not concern themselves with the origin of data.

3 Experimental Results

This project involves the use of two physical sensors and several modular algorithms to generate a single spatial representation of obstacles and free-space surrounding a mobile robot. The two physical sensors are a stereo pair of cameras and a single-axis laser range finder. The particular hardware used in these experiments were 2 SONY EVI-370 color CCD cameras and a SICK LMS-200 lidar. The "robot" used was JPL's IPN-cart, a two-wheeled non-powered cart with the cameras and lidar attached, and carrying on-board power supplies and computing.

The remainder of this section presents a snapshot of the system in action. Recall that each map contains an occupancy grid and a confidence grid: both are necessary to understand the interpretation of the map. In the scene illustrated, the full complement of sensors and signal processing modules is functional.

In this experiment, the robot is stationary in front of a rock, while a person is moving in from the lower right to the upper center of the map (directly in front of the robot, just to the right of the rock). In the time slice shown, the person is right of center near the bottom of the half-plane shown[3].

Figure 3 presents the raw lidar information while Figure 4 shows the lidar returns that survive filtering by the foliage detector. In this case, the foliage screener works very well, and the only three point-clusters that survive belong to the rock and the legs of the person. Figure 5 shows the stereo-derived range map, which shows the rock and some background foliage or ground, but does not pick up the person (the person is significantly away from the optical axis of the camera pair, a well-known failure mode for stereo-derived range [6]. Figures 6 and 7 show the summary data and confidence maps, respectively.

There are several things worth noting about the summary maps. These data illustrate the effect of fusing the data from several sensors with different measurement densities and extents. The confidence map clearly shows that confidence in data become zero at the limit of the sensor map from which these data are derived. In this case, the the lidar, foliage, and stereo sensors are all set to $10mm/$cell resolution with a 2 meter limit, but the tracking sensor is set to a $20mm/$cell resolution with a 4 meter limit. Therefore, the summary map has a greater extent than several of the component maps. It is cropped here for display purposes.

Referring again to Figure 7, there are four distinct confidence levels shown: (1) in several regions there is **very low confidence** in the data. These regions are typically regions behind obstacles, primarily lidar-blocking foliage in this case or regions outside the range of all of the sensors. (2) There are some data for which there are **low but nonzero confidence**. These data, the light grey regions located about 30-40 degrees from the horizontal, correspond to summary data from the previous frame that have not been reinforced in the current data. (3) There are also data that appear in multiple sensors, but with conflicting results. These appear as darker grey in the figure. Typically, these correspond to regions that the lidar asserts as obstacles but the foliage detector screens out (and therefore asserts as free space). These are marked as **free** in the data map, but **with somewhat lower confidence** than the next category. (4) The regions marked dark grey, a thin wedge at about 100 degrees and a larger wedge at 80-95 degrees, indicate data reinforced by multiple sensors. That is, several sensors agree on the occupancy status of the cells. These data have **high confidence**. This illustrates how the summarizer combines information from various sensors (including its previous map), using the rule base to weight the information to arrive at a reasonable composite that has better information than any of the component sensors.

Figure 8 illustrates the state of the object tracker. In this case, there are four point clusters that survive the thresholding. Two correspond to the rock and the person's legs as described above. The other two correspond to foliage behind and to the left and right of the rock, primarily introduced by the stereo range sensors. The dotted boxes correspond to the current observations (point clusters). The solid boxes correspond to current tracks. The apparent size of the vegetation behind and to the right of the rock leads to two independent tracks being created for this object, of different sizes. The observation for the person will update this track upward (away from the robot), and the rock track will remain stationary. All of this tracking is done from the summary map.

4 Conclusions

We have presented the architecture and some example results of a system that combines spatial occupancy information and associated confidences into a summary map. This makes it possible to design algorithms that work from this summary map and are relatively robust to sensor dropout and misbehavior, and can combine the reliable information from multiple sensors without building these capabilities into each and every module of the system.

Future improvements to this system might include increasing the sophistication of the object tracking

[3] The maps are actually symmetric about the robot, but as the set of sensors described in this paper are all forward-looking, only the front half-plane is displayed.

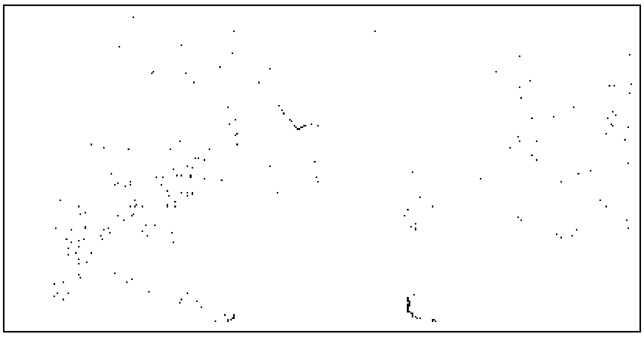

Figure 3: *Lidar Sensor Map*

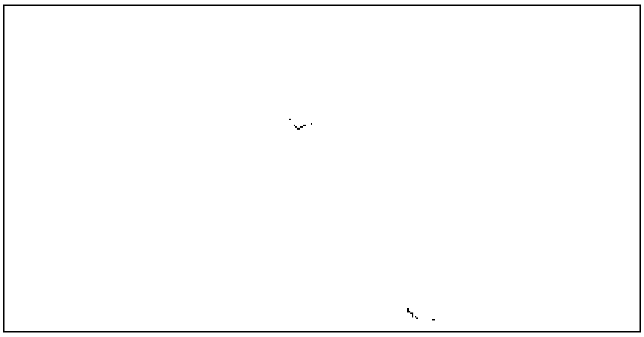

Figure 4: *Non-Foliage Sensor Map*

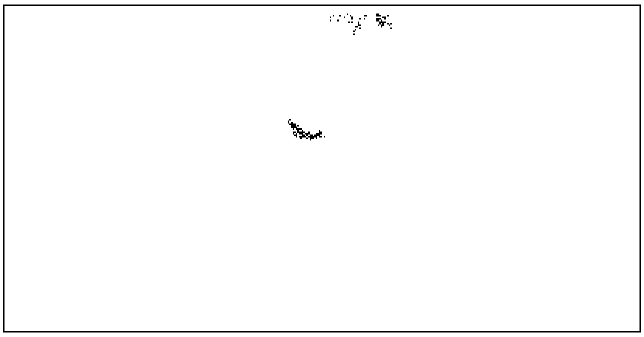

Figure 5: *Stereo-Derived Range Sensor Map*

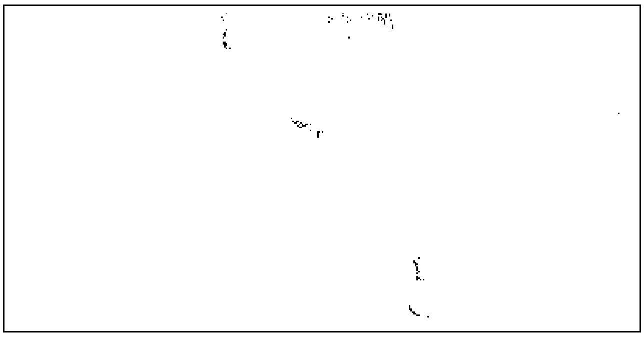

Figure 6: *Summary Data*

module to reduce the number of created but unconfirmed tracks. Also of interest would be better characterizing the various sensors self-estimate of their data reliability, and integrating this into a mobile robot platform such as JPL's URBIE urban mobile robot.

This system shows promise as a flexible sensor fusion framework for mobile robotics. Advantages to this system over existing systems include the ability to encapsulate sensors, so that downstream algorithms don't need to know the details of the sensor suite on the robot, and for the system to automatically adjust for conditions where a single sensor fails or performs poorly but other sensors are functioning properly, such as in open fields where the range of lidar data is quite limited or in grassy areas where stereo range has well-known problems.

Acknowledgments

The authors would like to thank NASA/ASEE Summer Faculty Fellowship program for making this work possible and JPL/California Institute of Technology for providing facilities and computing resources.

References

[1] P. Bellutta, R. Manduchi, L. Matthies, K. Owens, and A. Rankin. Terrain perception for demo iii. In *Proceedings IEEE International Conference Intelligent Vehicles*, Dearborn, MI, October 2000.

[2] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems.* Artech House, Norwood, MA, 1999.

[3] A. Castaño and L. Matthies. Foliage discrimination using a rotating ladar. In Progress, 2002.

[4] B. V. Dasarathy. *Decision Fusion.* IEEE Computer Society Press, Los Alamitos, CA, 1994.

[5] U. Handmann, G. Lorenz, T. Schnitger, and W. Seelen. Fusion of different sensors and alrogithms for segmentation. In *Proceedings IEEE International Conference Intelligent Vehicles*, 1998.

[6] E. P. Krotkov. *Active Computer Vision by Cooperative Focus and Stereo.* Springer-Verlag, New York, 1989.

[7] J. J. Leonard and H. F. Durrant-Whyte. *Directed sonar sensing for mobile robot navigation.* Kluwer Academic, Norwell, MA, 1992.

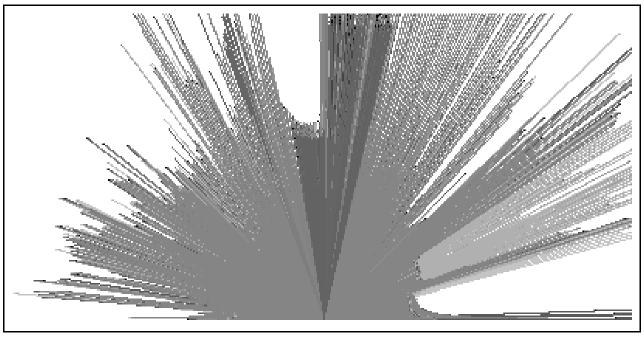

Figure 7: *Summary Confidence*

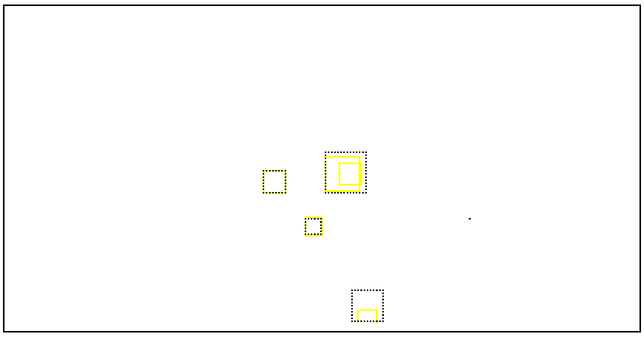

Figure 8: *Object Tracking Sensor Map*

[8] J. Macedo, R. Manduchi, and L. Matthies. Ladar-based discrimination of grass from obstacles for autonomous navigation. In *Proceedings International Symposium on Experimental Robotics*, Honolulu, HA, December 2000.