A Contour Method for Real-Time Range Image Parsing

Nicolas Chauvin[1] Gaetan Marti[1] Kurt Konolige[2]

[1]VRAI-Group / DMT-IMT
École Polytechnique Fédérale de Lausanne
CH-1015 Lausanne

[2]SRI International
333 Ravenswood Ave
Menlo Park, CA 94025

Abstract

Real-time or near real-time range images are available from a variety of sources. Many proposed uses for these devices require significant post-processing to extract relevant 3-D information. We propose a new method based on a generalization of *contour maps* that has several advantages over current methods in tasks such as terrain reconstruction and obstacle avoidance. The method is illustrated with examples from obstacle avoidance on a mobile robot.

1 Introduction

Current range imagers have achieved real-time or near real-time performance on images of modest size. For example, stereo algorithms on standard hardware are capable of returning dense 320x240 range images video rates, while scanning laser range-finders can operate at 2 Hz on 256 x 256 images [Konolige 1997, Woodfill and Von Herzen 1997].

To take advantage of these devices, researchers have proposed several methods for post-processing the range images to extract 3-D information. We can classify them into two basic types :

- Grid-based methods. Here the 3-D space is discretized using regular grids, and points in the range image are transformed to the corresponding Cartesian grid locations [Stentz and Hebert 1995].
- Surface interpolation methods. Points in the range image are transformed into points in 3-D space, and then triangles or other surface patches are interpolated between the points [Faugeras 1993, Buffa *et al.* 1993].

As noted, these methods all operate by transforming the range images into representations in the 3-D Cartesian space: *volumetric* representations. While this is the most direct method of extracting 3-D information, it has several disadvantages. First, the representation is changed from a 2-D range image to volumetric space, increasing the amount of data to be dealt with. This is especially true for surface interpolation methods, since the representation and manipulation of 3-D surface patches is costly. As a consequence, these methods are not currently being used in real-time interpretation.

Methods based on 3-D grids divide space using a regular 2-D grid on the ground plane, and transform points from the range image into the appropriate grid. Further quantization in elevation is introduced in the *occupancy grid* model. While the complexity of grid-based methods makes them more amenable to real-time processing, there is difficulty using grid representations to extract analytic models that are useful. For example, in finding obstacles for a typical outdoor robot, one might test (for example) slope changes greater than 30 degrees over 1 meter. In a volumetric representation, finding such obstacles involves search over a large volume, rather than the simpler 2-D representation of the original range image.

Finally, volumetric representations over regular grids may not reflect the precision and error bounds of the range images. The range precision of stereo images falls off radically with distance, and any range method has elevation errors that grow with distance, based on geometry of the device viewpoint.

For many applications, working in the Euclidean space turns out to be unnecessary and difficult to manage. In order to reduce the amount of data that has to be processed, we introduce a method of quantifying volumes that allows us to manipulate range images more directly, without having to first transform to 3-D space. The method is inspired by the standard use of contour maps to represent elevations; hence, we call it the *contour method*. A contour represents the elevation at a particular height; all terrain between one contour line and the next is at an elevation between that represented by the contour lines. Contour creation can be visualized as a set of planes parallel to the ground at specified heights, intersecting the terrain. These *cutting planes* induce a quantization of the 3-D space based on elevation.

Our approach generalizes this basic idea, and consists of the following steps:

1. Constructing a set of volumes in 3-D space using a set of cutting surfaces (not necessarily planar).

2. Projecting the cutting surfaces back to the range image to induce a quantization of the range data.
3. Using the quantized range image to construct terrain models or other abstractions.

In cases where the desired segmentation is relative to the sensor viewpoint, the first two steps can be achieved off-line, leading to significant computational savings, especially when the cutting surfaces are complicated. In addition, step 3 can often be performed in the range image space, which is much more efficient than working in the volumetric space. Also, in contrast to the grid-based approaches, the cutting surfaces need not be regular, and can be sized to take into account the precision and error characteristics of the range data.

The advantage of using the contour method is that expensive calculations in the 3-D space do not have to be performed. Instead, all calculation takes place on a pixel-by-pixel basis in the range image space. The end result of our contour method is a realtime algorithm for range image segmentation that operates in the range image space. The algorithm is proportional to the size of the range image, and in general is very efficient on standard processors. For example, we have implemented a realtime obstacle avoidance system, running at 30 Hz, using stereo images.

The next section presents the notation and mathematics of projective geometry relevant to the contour method. Section 3 gives an example of the use of the method in obstacle avoidance, implemented on a small mobile robot, Pioneer I.

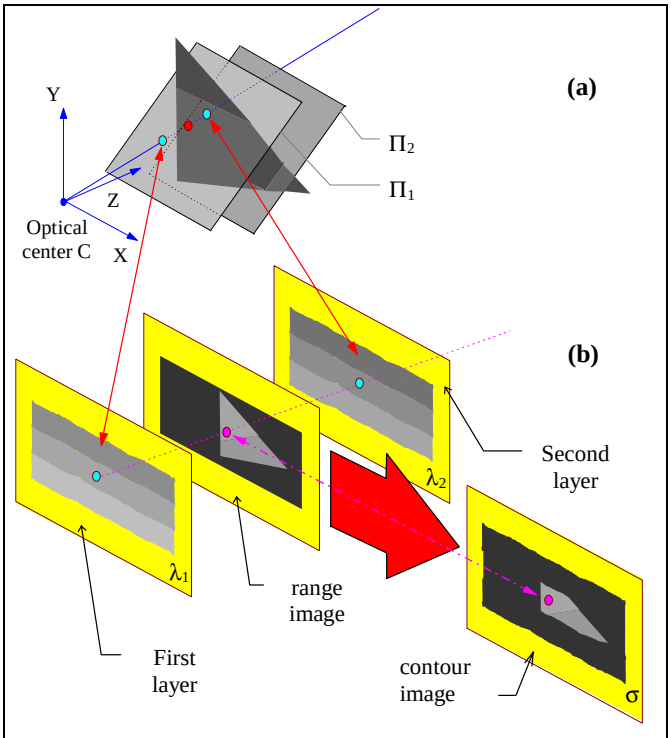

Figure 1 *Schematic representation of the contour method. (a): object intersecting cutting surfaces (b) corresponding layers in the image plane and representation of a co ntour.*

2 Layers and Contours

Figure 1 shows the basics of the contour method. A set of cutting surfaces Π_i define volumes in 3-D space (a). The volume is called a *slice*. In the figure, the slice is the volume between the two surfaces Π_1 and Π_2. Where the slice intersects an object, it determines the portion of the object lying between the surfaces.

The portion of the object within a slice has a corresponding representation in the range image, formed in the following way. When the cutting surfaces are projected onto the range image plane in (b), they are called *layers*. Parts of the range image that lie between two layers in terms of their range measurement must be in the corresponding slice.

The result of segmenting the range image between two layers is a *contour* σ, the representation of the object slice in the range image space. The pink point is in this slice, hence in the segmented image contour.

Given that the cutting surfaces are defined relative to the optical center *C*, their projection into layers can be done off-line. The only computation needed for the extraction of contours is a point-by-point comparison between the range image values and those of the layers.

2.1 Surface projection to layers

Here we derive the formulas for constructing layers from 3-D cutting surfaces. Consider a point $M = [X_C, Y_C, Z_C]^T$ belonging to $\Re^3$ (see Figure 2). Its projection in the *two-dimensional image space* is *m* belonging to $\Re^2$, where $m = [x_c, y_c]^T$.

For the special case where c is on the optical axis defined by $\boldsymbol{Z}$, the following relation is valid:

$$\begin{cases} x_c = f \cdot \dfrac{X_C}{Z_C} \\ y_c = f \cdot \dfrac{Y_C}{Z_C} \end{cases} \qquad (1)$$

A *layer* is a range image produced from the projection of a surface in $\Re^3$ via the pinhole model just described. A set of surfaces must have the following properties to be considered valid cutting surfaces :

(i) The different surfaces must be non-intersecting (as the *equipotentials* of a field).

(ii) An optical ray l must intersect the surface at most once.

Given the pinhole geometry of Figure 2, it's a simple matter to calculate the layer corresponding to a cutting surface. First, we define a *range image* as an image r where the value of the image $r(m)$ at any point m is the distance along the Z axis to the corresponding point M. The range of a given image point may be undefined, in which case $r(m)$ is set arbitrarily to 0. The range metric is a function of the depth coordinate Z_C of the 3-D image of m : $r(m) = f(Z_M)$. In general any monotonic function f is allowed ; usually we choose it to an appropriate metric for the range device, e.g., *disparity* for stereo imagers.

The *layer* of a cutting surface Π_i, denoted λ_i, is a range image of the surface Π_i. Condition (ii) above guarantees that a cutting surface has a unique projection to the range image, although the range may not be defined everywhere on the image. Figure 3 shows the projected layer of a surface on the image plane.

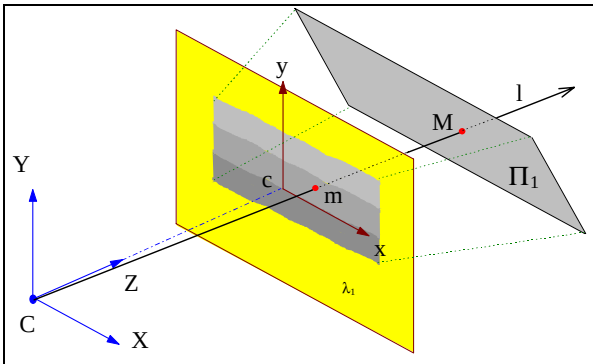

Figure 3 *Partial representation of the surface Π_1 and its corresponding layer λ_1 in the range image frame. Note that in this example, the range function r is discrete.*

Given two layers, a point of the range image that lies between the ranges of these layers is guaranteed to lie in the volume delimited by the two cutting surfaces of the layers. To state this more formally, we define the volume between two cutting surfaces (a *slice*). A point M belongs to a slice V bounded by the cutting surfaces Π_1 and Π_2 if the following condition holds (see Figure 4) :

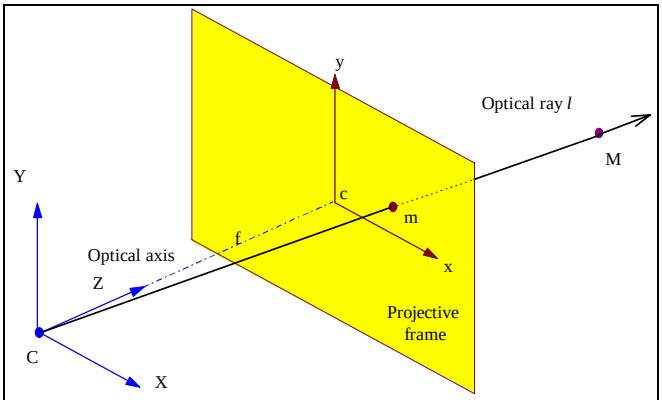

Figure 2 *Pinhole model: image of the point M on the projective plane. C denotes the optical center of the image, f the focal distance. The coordinate system relative to the three-dimensional space is (C, X, Y, Z) and the one relative to the projective space is (c, x, y).*

$$M \in V \Leftrightarrow \exists M_1 \exists M_2 \,\big|\, \Delta(C, M_1) < \Delta(C, M) \le \Delta(C, M_2) \qquad (2)$$

where $\Delta(A, B)$ is the Euclidean distance between point A and B. A point m of a range image belongs to the *slice* $\sigma = \sigma_{1,2}$ delimited by two layers if :

$$m \in \sigma_{1,2} \Leftrightarrow \exists \lambda_1 \exists \lambda_2 \,\big|\, r_1(m) < r(m) \le r_2(m) \qquad (3)$$

Moreover :

$$m \in \sigma_{1,2} \Leftrightarrow M \in V \qquad (4)$$

Note that a contour is a segmentation of the original range image, so that the range values are kept in the contour. This can be useful for further processing, e.g., mapping the contour onto 3-D space.

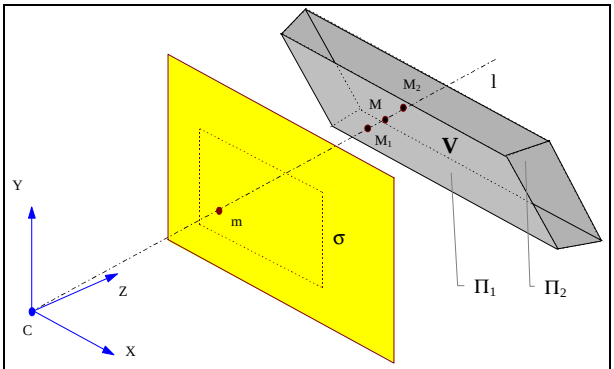

Figure 4 *Representation of a point m in the contour σ (see text). Note that if the point M belongs to the slice V then m belongs to the contour σ.*

2.2 Elevation Contour Example

Figure 5 shows typical results using a disparity image from a stereo ranging system, and segmenting the range image using cutting planes at different elevations from the ground plane. The disparity image (a) is in the upper-left corner: larger disparities are lighter, and indicate surfaces closer to the optic center. The stereo device is placed so that the optical axis of the cameras are parallel to the ground and about 22 cm above it. A sequence of 9 planes parallel to the ground at different elevations are used as cutting surfaces to segment the range image into contours; some of the contours at different elevations are shown in (c). Successive contours pick out the portions of objects that are at different heights relative to the ground plane. In the top right image (b), these contours are collected to show the heights of obstacles in front of the cameras. Note that this is *not* a range image; the original range image has been segmented according to the contours, and the ground plane contour is assigned a low brightness value, while portions of the scene above the ground plane are assigned successively higher brightness values. The shape of the obstacles in terms of their elevation from the ground plane, which is not evident from the initial range image, becomes very clear.

Computationally, the construction of the contours, as well as the combined elevation image, is inexpensive. Let N be the number of points in the range image, and K the number of layers. We pre-compute the layers, and at each point m in the range image we associate an ordered sequence of numbers $l_1(m), \ldots l_K(m)$ which are the range values of the contours at that point. Since searching a sequence can be done in $O(\ln K)$, the time needed to construct a contour or the combined contour image is $O(N \ln K)$.

In some cases, like the VFH application below, constructing a single contour is sufficient, and subsequent processing for the application can also be done in the range image, and adds little to computational burden. In other cases, further processing of contours must take place, sometimes by transforming them to the 3-D space.

Although this example used cutting planes, in principle the contour method presented here is applicable to arbitrary convex volumes. For example, if there is a particular volume in which we wish to detect intruders, then it suffices to depict that volume by its bounding surfaces, and map these back into contours in the range image.

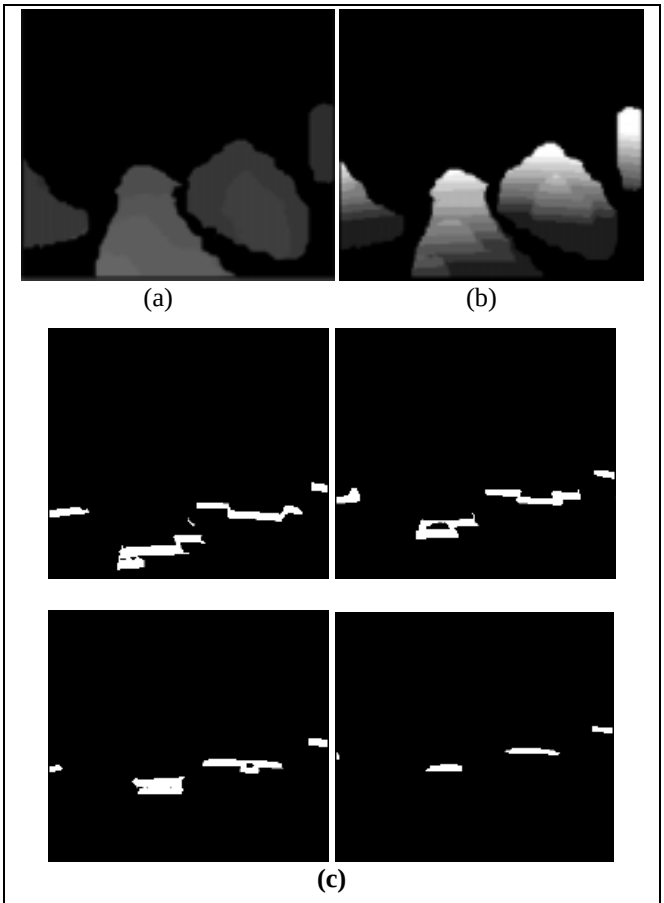

Figure 5 *Contour method used with horizontal slices of different elevations. Image (a) represents the range image, (b) the elevation map and (c) different layers.*

3 Application to Obstacle Avoidance

We have tested the contour method on a small mobile robot (Pioneer I), using a real-time stereo vision sensor [Konolige 1997]. The purpose was to avoid obstacles, using stereo range images processed at 30 Hz. The stereo images were sent by video wireless to a PC, where 160x120 range images were computed, and input to the contour method algorithms. These algorithms provide information about the location and range of obstacles and holes, which can be used for obstacle avoidance.

A suitable method for obstacle avoidance is the *Vector Field Histogram* (VFH) method of [Borenstein and Koren 1991]. We describe the VFH algorithm, and the contour method processing for detecting obstacles and holes.

3.1 Obstacle avoidance using VFH

As originally developed with sonar sensors, the VFH method employed three steps:

1. A regular 2-D *histogram grid* in plan view, holding the results of sonar sensor readings around the robot. The value of each grid point represents the number of sonar readings that indicated an object within the point.
2. A *polar histogram* constructed from the histogram grid, with k regular angular sectors instead of a rectilinear grid. The value h_k of each sector in the polar histogram represents the obstacle density in that direction.
3. Steering and velocity values extracted from the polar histogram.

In range images, each column of the image represents a polar sector whose angular width is determined by the camera parameters. We let the k sectors correspond to the columns of the range image. Thus, we can construct the polar histogram directly from the contour representation, without having to convert to Cartesian space.

The key step is calculating the histogram value h_k for each sector. Roughly speaking, this value represents the probability of finding an obstacle close to the robot in the direction of sector k. The simplest idea is to use a single cutting surface at elevation over the ground plane sufficient to constitute an obstacle for the robot. Any points in the resulting contour are obstacles, and we can use the number of such points in a column and their distance to determine a histogram value.

The details of the weighting scheme we use are not critical; we expect almost any reasonable method that combines distance and number of points will work reasonably well. Since in the implementation we used a stereo system with disparity as the range metric, each contour cell m contributes $\ln[r(m)]$ to its sector value. This measure compensate for the fact that the disparity goes up hyperbolically as an object gets closer.

Error! Reference source not found. shows a calculation of the polar histogram from a typical range image and a single-contour segmentation (top and row (b)). In this case, the sensor covers about a 70 degree angle, and each sector is about 0.5 degrees. In the top row, the left image is an intensity image of the scene, and the middle one shows the disparity map computed by a stereo system. Brighter green values are higher disparities, hence closer to the camera. The right image is a contour map, computed with a series of planes parallel to the floor, from 10cm to 50cm. The red areas are those closer to the floor, while green are higher elevation contours.

In row (b), we compute the VFH using a simple characterization of obstacles as anything object from 10cm to 20cm in height (the robot is about 20cm high). The contour image (left image) picks out the object surfaces that are in this volume, and the VFH (right image) shows the clear directions of travel, with the optimal direction as a vertical green line. Because of the small sector size, there is considerable variation in adjacent histogram values, and the result could benefit from low-pass filtering.

Row (c) shows the flexibility of the contour method in deciding obstacles under different conditions. Here, the height is chosen to be 15cm to 40cm, effectively including obstacles that may represent "overhanging" surfaces for a small robot. The contour image is on the left, and the VFH on the right. The red color codes indicate areas that are dangerous to traverse because of their height.

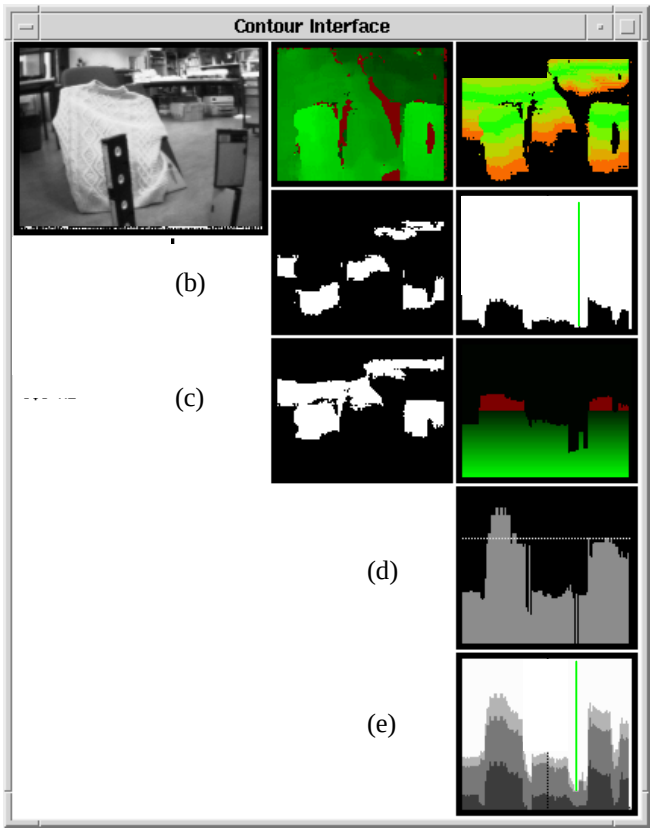

Figure 6. VFH results using the contour method. The upper row shows the input grayscale image, the disparity image, and a contour parsing using planes parallel to the floor. Row (b) shows obstacles from 10cm to 15cm above the floor, and the resultant VFH, with the selected direction. Row © shows obstacles from 20cm to 40cm in height, and the VFH in color, with red areas being dangerous. Row (d) is a polor histogram of mean elevation angles for a particular poar direction; the line is at 80 degrees. Row (e) shows a weighting of all these factors, with the resultant best travel direction as a vertical green line.

3.2 Obstacle Elevation Angles

Additional sophistication in the construction of the polar histogram from contours is possible with the contour method.

1. Ground plane detection. A contour representing the ground plane would give an indication that there actually was a reasonable path in front of the robot. Here, the value $\ln[r(m)]$ for each cell in a sector would be *subtracted* from some initial constant for the sector.
2. Holes. A contour underneath the ground plane could be used to check for holes near the robot. The addition to the histogram would be the same as for positive-elevation obstacles.
3. Small-height obstacles. Instead of a single contour at an appropriate height for obstacles, several could be positioned starting from just over the ground plane. The contribution of elements from the lower contours would be weighted by a fraction q depending on their height. Thus, the robot would prefer smooth terrain to bumpy, even though it could negotiate the latter.
4. Obstacle elevation angle. It's not necessarily dangerous for a robot to go above the ground plane, given that the rise in elevation is gentle. Instead of using the absolute distance above the ground plane, we could use the angle at which the ground rises.

We have implemented this last idea using the contours already described. A post-processing step is performed, where the elevation angle is estimated by looking at the vertical thickness of each contour. The elevation angle is a function of this thickness, and can be readily computed. Row (d) shows the mean elevation angle computed in this manner, for each of the directions in the polar histogram. The horizontal line represents an angle of 80 degrees.

Finally, row (e) shows a weighting of all the obstacle types in a combined VFH. The optimal direction of travel is given by the vertical green line.

3.3 Implementation

The VFH method was implemented on a standard PC (233 MHz Pentium II), computing stereo images and the VFH algorithm. The disparity images were 160x120 at 16 disparities. The VFH processing took less than 10 ms per image to format the polar histogram and extract the desired direction of travel. The whole system runs at video rates, using only half the computing resources of the PC.

.

4 Conclusions

We have defined a method for processing range images based on *contours*: a segmentation of the images induced by the projection of 3-D surfaces. The method uses the idea of elevation contours familiar from map-making, but generalizes it to arbitrary nonintersecting 3-D surfaces. We have presented some results, including an implemented system for real-time obstacle avoidance using the VFH method. The computational resources required for extracting contours are modest, and the VFH application runs at video rates.

5 References

[Borenstein and Koren 1991] J. Borenstein and Y. Koren. The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots. *IEEE Journal of Robotics and Automation* **7**(3), June 1991.

[Buffa *et al.*, 1993] M. Buffa, O. Faugeras and Z. Zhang. A stereovision-based navigation system for a mobile robot. *Rapport de Recherche N°. 1895*, INRIA, Sophia-Antipolis, April 1993.

[Faugeras 1993] O. Faugeras. Three-Dimensional Computer Vision: a Geometric Viewpoint. MIT Press, 1993.

[Konolige 1997] K. Konolige. Small vision systems: algorithms and implementation. *ISRR97*, Hayama, Japan , October 1997.

[Stentz and Hebert 1995] A. Stentz and M. Hebert. A Complete Navigation System for Goal Acquisition in Unknown Environments. *Autonomous Robots*, 2, 127-145, 1995.

[Woodfill 1997] J. Woodfill and B. Von Herzen. Real-Time Stereo Vision on the PARTS Reconfigurable Computer. *IEEE Symposium on FPGAs for Custom Computing Machines*, April 1997.