Estimating the pose of 2-d objects using potential functions and quadtrees

[1]Régis Houde, [2]Jacques Tremblay, [2]Denis Laurendeau, [1]Michel Pelletier

[1]Institut de recherche d'Hydro-Québec
Varennes, (Qué), Canada, J3X 1S1
regie@ireq-robot.hydro.qc.ca

[2]Laboratoire de vision et systèmes numériques
Dép. de génie électrique, Université Laval
Québec, (Qué), Canada, G1K 7P4
laurend@gel.ulaval.ca

Abstract

This paper presents an approach for the estimation of the pose of objects in 2-D images. The technique relies on the combined use of potential functions and quadtrees to compute candidate poses. The quadtree representation of the image is used to generate rough pose estimates that are then iteratively refined. The final pose estimate is validated by a human operator that supervises the pose determination system. Examples of pose estimation are presented for both synthetic and actual image data. The method will be extended to 3-D object by using an octree model of the scene.

1 . Introduction

A common task in the application of computer vision to robotics is to estimate the pose (position and orientation) of an object in a scene. Pose determination is a sub-problem of the more general problem of object recognition and may be formulated as computing the parameters of the rigid transformation that is needed to bring an object from a given canonical position to its actual position and orientation in the scene.

In the past, a number of different approaches have been proposed to solve the pose determination problem. These approaches differ in the number and type of assumptions made to reach a solution. For instance, model-based approaches try to match features detected in the image with corresponding features on a model. However, it is often difficult to search the image for specific low-level features. Models are typically simple and do not encode a high-level representation of the objects. When the image features are simple (point, lines, corners, etc) but numerous, the matching process may be time consuming. Furthermore, matching errors between image and model features result in poor pose estimates.

Grimson and Lozano-Perez have introduced the *interpretation tree* paradigm for object recognition and localization. The method operates by examining all hypotheses about pairings between image features and model features. Inconsistent pairings are discarded by using local constraints on the distance between pairs of image features and their corresponding model features. Large parts of the tree may be eliminated from very simple geometric constraints such as the distance between points or the angle between surface normals [3]. Several pose candidates may remain at the end of the matching process and the best one must be selected. This is often done by using a least-squares technique. Archibald *et al* have proposed a similar approach using edges extracted from sparse 3-D range images [1].

Object recognition and localization is also possible via pose clustering [6]. This approach, which may be considered as a generalyzed Hough transform, is characterized by an accumulation of low-level evidence followed by a clustering step which selects pose hypotheses with strong support from the set of evidences (maxima in the clustering space). The clustering step requires that a pose estimate be computed for each low-level feature match. This is time consuming since only a small subset of these pose candidates will be considered at the end of the process.

The geometric hashing technique avoids the computation of multiple pose estimates by an intensive off-line model preprocessing or learning stage where model information is indexed into a hash-table using minimal, transformation invariant features [4]. Pose is computed only for those matches which contain the most evidence and which are chosen from an on-line voting scheme. Since most of the computation is performed off-line, geometric hashing shows numerous advantages over other techniques for real-time applications. However, several pose candidates still have to be tested before the best one is found.

This paper describes a model-based pose determination technique using potential functions and quadtrees. It is assumed that the set of possible objects that may appear in the image is known *a priori* and that a description of each object is available in a library of models. The pose estimation process is supervised by a human operator which brings an instance of the model near its occurrence in the image and the algorithm refines this rough estimate by moving the model until the final pose is found by

minimizing a potential function. A quadtree encoding of the scene is used to reduce the computation time of the potential function in the first iterations of the pose determination process. Section 2. presents the basic idea of pose estimation using potential functions and presents the type of potential function that is used by the algorithm. Section 3. describes how the computation time of the potential function may be reduced by modelling the scene with a quadtree. Section 4. presents experimental results. Section 5. concludes on how the technique can be extended to 3-D scenes.

2 . Pose estimation using potential functions

2.1Basic principle of pose estimation using a potential function

Let us imagine a 2-D intensity image of a scene composed of several objects. The gradient of the image is computed and the resulting edge image is thresholded to produce a binary image such as the one shown in Figure 1.

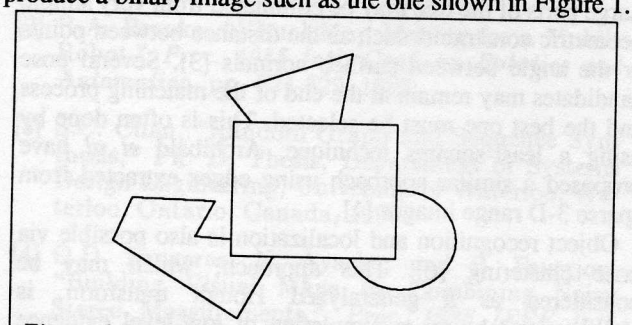

Figure 1 Typical binary image

Let us now consider each point of the binary image as a *positively charged electrical particle*. The charge induces an electrostatic field in the surrounding space. The expression of this electrostatic field is simply:

$$\vec{E}(\vec{r}) = \frac{1}{|r|^2}\vec{u}_r \qquad (1)$$

The electrostatic potential at a distance r_i of the point is obtained from (1):

$$\phi(r_i) = -\int_{\infty}^{r_i} (\vec{E}\cdot d\vec{l}) = -\int_{\infty}^{r_i} \frac{1}{r^2} d\vec{r} = \frac{1}{r_i} \qquad (2)$$

The total electrostatic potential at point (x, y) resulting from the combined effect of all points of the binary image is:

$$\Phi(x, y) = \sum_i \frac{1}{r_i} = \sum_i \frac{1}{\sqrt{(x-x_i)^2 + (y-y_i)^2}} \qquad (3)$$

The potential function in (3) is not very practical since it takes an infinite value for each (x_i, y_i). However the following function:

$$\Theta(x, y) = \frac{1}{\Phi(x, y)} \qquad (4)$$

is well-behaved in the neighborhood of image points. Figure 2 shows an example of the function $\Theta(x, y)$ that is produced by four colinear points in a 1-D scene. Figure

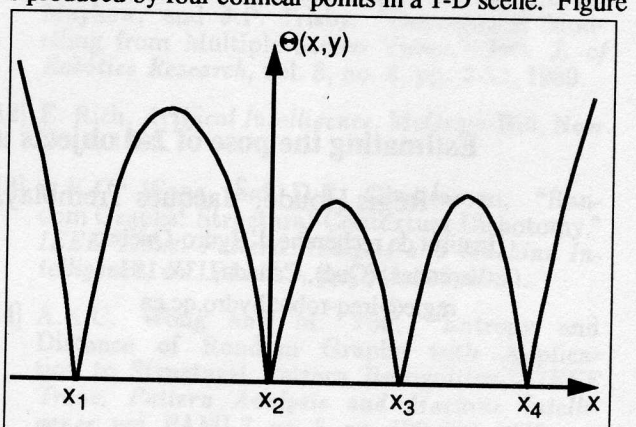

Figure 2 Potential function $\Theta(x, y)$ created by four colinear points.

2 shows a distribution of potential created by four points in a plane.

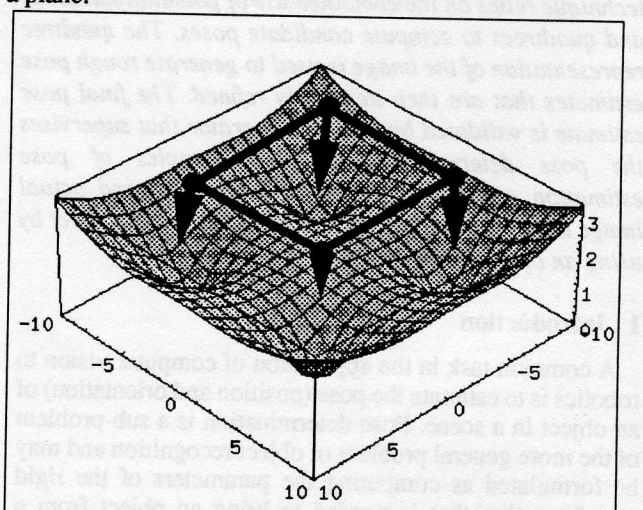

Figure 3Potential function $\Theta(x, y)$ created by four points located at (-5, -5),(-5, 5),(5, -5), and (5, 5) in the plane.

Let us assume that the set of four points in Figure 2 represents the occurrence in the image of a square-shaped model. We are interested in finding the pose (two translations and one rotation) of this square in the image relative to a global reference frame. The pose estimation is guided by a human operator. A convenient way to estimate the position and orientation parameters is to have the operator drag a copy of the square-shaped model in the vicinity of the instance of the square in the image. By assigning a *negative charge* to each corner of the model, the electrostatic force, which is along the direction of the gradient of the potential function produced by the image, will pull this model toward the bottom of the potential well in order to lower its energy. At equilibrium, we claim that the the model will match the pose of its image and that it is

thus possible to recover the pose parameters from the match since the rigid transformation between the model's initial and final positions are known. Figure 2 shows a schematic diagram of the several steps of the pose estimation method.

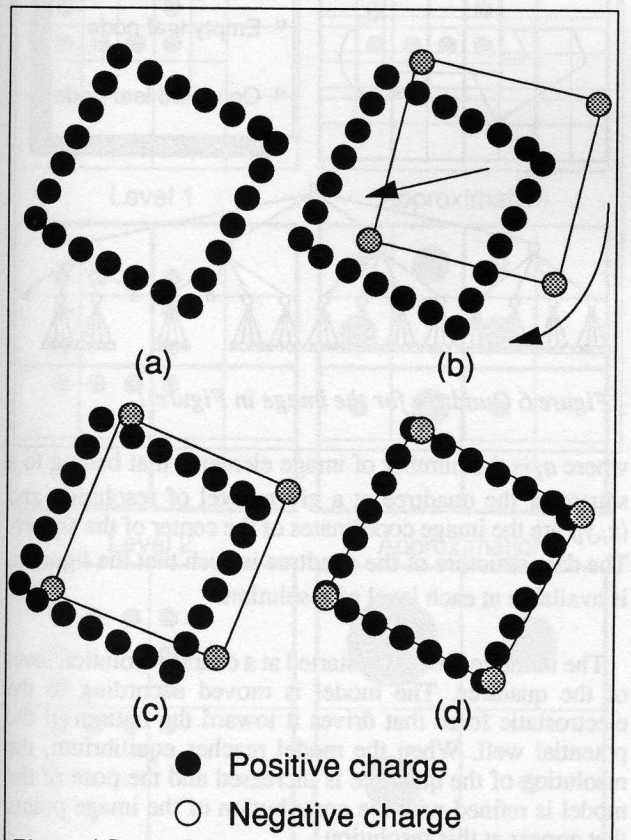

Figure 4 Steps of the pose determination approach for a simple square-shaped model. An instance of a square in the image is shown in (a). The image points (dark dots) are positively charged and produce an electric field in the "image space". In (b), a square-shaped model is dragged into the image, near the instance of the square. The corners of the model are negatively charged and are thus submitted to a force that pulls them toward the bottom of the potential well (similar to the one shown in Figure 2) as depicted in (c). In (d), the two objects are aligned and the pose of the object in the image may be computed from the rigid transformation that was required to bring the model to its final position.

2.2 Various steps of the pose estimation approach

The basic principle of pose estimation using a potential function has been presented in the previous section. We now describe each step in detail. In the following, it is assumed that the image has been segmented into edges and that a binary image has been obtained from the edge image. Several algorithms may be used for edge detection (for example, see Canny's approach in [2]). The image may contain several different objects that are allowed to overlap. Let us recall that a description of each object that may appear in the scene is available in a pre-computed library of models.

The steps of the pose determination algorithm are:

1- consider that a positively charged particle is attached to every edge point in the binary image. These charges create a potential function described by Eq (4).

2- an operator, whose role is to supervise the pose determination operation, say in a telerobotic application, locates an instance of an object in the thresholded edge image of the scene. The instance of the object may be incomplete, meaning that other objects may partially overlap its contour in the image. The operator then selects the computer model of the object stored in a library and drags this model near its occurrence in the scene. He then starts the pose estimation procedure and waits for the result. In the present application, the structure of the model is simply a set of characteristic points located on its constituent edges.

3- Once the pose determination procedure has started, the program computes the resulting force that is imposed on the model by the distribution of charge in the binary edge image. The model is allowed to move one step in the direction of the force. The size of the step must be chosen carefully. If it is chosen to be proportional to the gradient of the potential function at the current location of the model in the image, the pose estimation algorithm will tend to be unstable since the gradient takes very large values near image points, which means that the step increases significantly as the model moves toward the bottom of the potential well. For the present application, we have chosen the step to be proportional to the average value of the potential at the current position of the model. Since the potential function in (4) decreases near image points, the size of the steps also tends to decrease as the model reaches the bottom of the potential well, thus avoiding large steps near the bottom of the potential well which could affect convergence.

   The model is allowed to move step by step (the electrostatic force being computed at each step) until its potential energy reaches a minimum which is validated by the operator.

4- Once the model has reached equilibrium, the parameters of the rigid transformation that was required to bring it from its starting position to its final position are computed.

3 . Using the quadtree to approximate the potential function

Even though the pose determination approach described above is conceptually simple, its practical implementation raises several difficulties. The electrostatic force on each point of the model must be computed at each step of the algorithm. The computation of this force requires the

computation of the gradient of the potential. A convenient approach to compute this gradient is to find the tangent plane to the potential function at the point of interest. The direction of the gradient rests along the direction of the normal as shown in Figure 5. However, when there is a

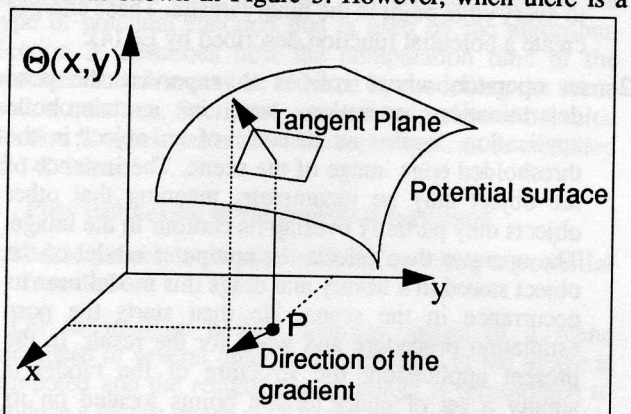

Figure 5 Computation of the gradient of the potential function Θ(x, y) at a given point P.

large number of edge points in the image and several model points, it is time consuming to compute the potential function (and consequently the gradient) for each point of the model. Since a high accuracy in the displacement of the model toward the bottom of the potential well is not required at the first steps of the algorithm, it would be wise to use a rough approximation of the potential function for these first steps and then to refine the function as the model gets closer to its equilibrium position.

A quadtree representation of the binary image was adopted to approximate the potential function [5]. The quadtree is very appealing because of its hierarchical structure which allows to represent the image at several levels of resolution. The quadtree recursively divides the image into square areas which are assigned a label according to the state of the image in the square. The *empty* label is assigned to squares which do not contain any image point. The *occupied* label is assigned to the square if there is at least one point in the square. Figure 1 gives an example of a quadtree for the image in Figure 1.

The pre-defined size of the smallest square defines the resolution (e.g. number of levels) of the quadtree. Of course, it is useless to choose a resolution for the quadtree that is smaller than the pixel size (or, for that matter, the resolution of the image sensing device)

The reduction of the complexity for the computation of the electrostatic force on the model by the image points is accomplished as follows. In the first steps of the iterative process, all image points belonging to a given square at a given level of the quadtree are considered to be located at the center of mass of the square and the equation for the computation of the potential at point (x,y) of the image space reduces to:

$$\Phi(x,y) = \left(\sum_i \frac{a_i}{\sqrt{(x-x_i)^2 + (y-y_i)^2}} \right)^{-1} \quad (5)$$

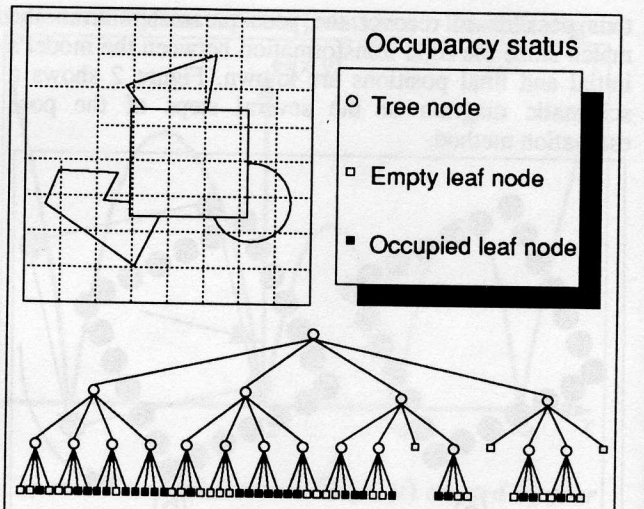

Figure 6 Quadtree for the image in Figure 1.

where a_i is the number of image elements that belong to a square of the quadtree at a given level of resolution and (x_i,y_i) are the image coordinates of the center of the square. The data structure of the quadtree is such that the figure a_i is available at each level of resolution.

The iterative process is started at a coarse resolution level of the quadtree. The model is moved according to the electrostatic force that drives it toward the bottom of the potential well. When the model reaches equilibrium, the resolution of the quadtree is increased and the pose of the model is refined with the contribution of the image points that appear at this resolution.

4 . Experimental Results

The pose estimation technique presented above has been applied to several scenes of varying complexity. Figure 8 shows the electrical field generated by a single rectangular shape while Figure 9 shows the field generated by a complex scene made of several overlapping shapes.

Figure 10 shows the pose determination in progress for the half-circle in the upper part of Figure 9. The short arrows indicate the direction of the electrical force that is imposed on each point of the model while the larger arrow shows the direction of the resulting force on the center of mass of the model. The model is moved in the direction of this force by a step that is proportional to the average value of the potential at the current position of the model, as mentionned at item 3- in Section 2.2

The binary edge image in Figure 10 was generated by computer to check the behavior of the pose estimation technique. Figure 11 shows the results of the application of the algorithm on actual 2-D data representing the outline of a wrench..

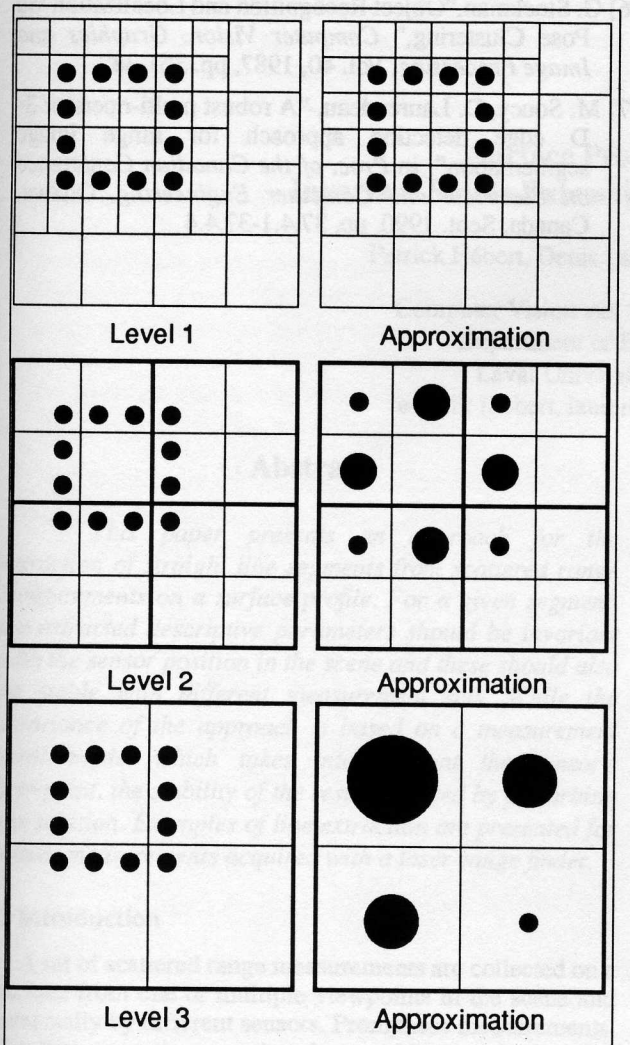

Figure 7 Three levels of a quadtree are shown on the left, along with the approximation that is used for the computation of the potential function on the right. The size of the dot is proportional to the number of points at the highest level of resolution of the quadtree.

The pose estimation algorithm was started at the 5^{th} level of the octree (e.g. each square of the quadtree has length 2^5 pixels). The initial pose of the model was: x=250, y= 276, θ= -10.1º. The final pose estimated by the algorithm was, after the matching between the model and the image has occured, x= 273.8, y= 292.2, θ= -0.1º. A total of 96 iterations was required to reach the final pose estimate.

5 . Conclusion

This paper has presented a pose estimation technique based on potential functions and the quadtree model of the scene. The results are very good for synthetic data and the pose is found with good accuracy (within pixel resolution) in acceptable computing time. For actual images with a given noise level, the pose estimates are also satisfactory but a small error remains in the estimate since noisy edge images do not allow to fit the model perfectly into the

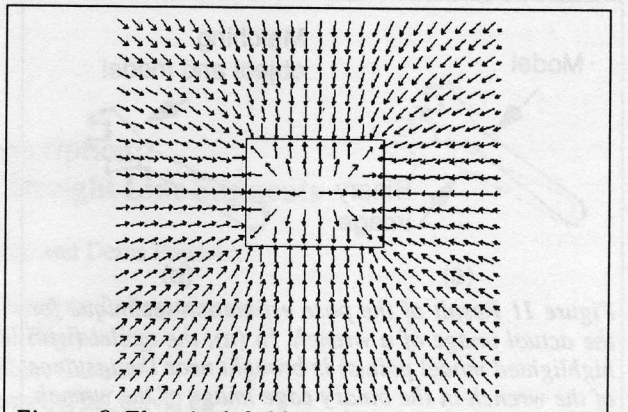

Figure 8 Electrical field produced by a rectangular shape in the binary edge image of the scene. This field drives the model towards the bottom of the potential field associated with the field.

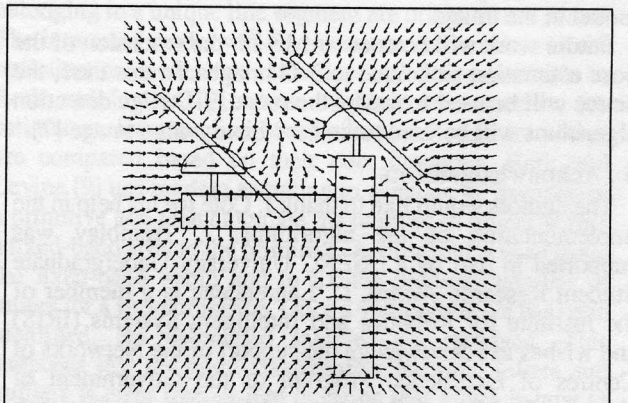

Figure 9 Electrical field produced by several objects of complex shapes in the binary edge image. It is desired to find the pose of one of these objects using the potential function and the quadtree of the scene.

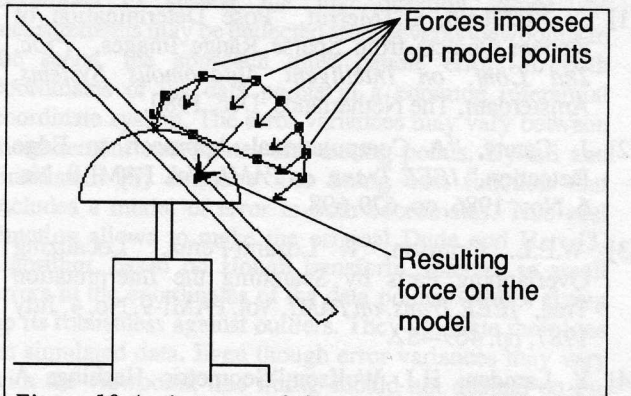

Figure 10 An instance of the model of a half-circle is pulled toward the half-circle shape in the image by the force imposed on its feature points and produced by the electrical field generated by the points of the binary edge image. This force is computed for a given resolution of the quadtree modelling the scene.

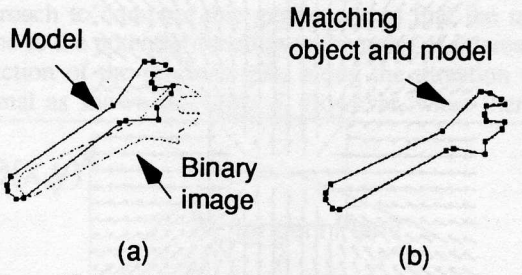

Figure 11 Result of the pose estimation technique for the actual image of a wrench. In (a), the model (with highlighted model points) is brought near the instance of the wrench in the binary edge image of the wrench. In (b), the matching between the model and the image is completed following the application of the pose estimation algorithm.

bottom of the potential well created by the instance of the model in the image.

Future work will be concerned with the extension of the pose estimation technique to 3-D images. In this case, the octree will be used to model the scene. 3-D edge detection algorithms will be used to find the binary edge image [7].

6 . Acknowledgements

The authors would like to thank J. Côté for his help in the implementation of the algorithms. J. Tremblay was supported in part by a NSERC University Undergraduate Student Research Award. D. Laurendeau is a member of the Institute for Robotics and Intelligent Systems (IRIS) and wishes to acknowledge the support of the Networks of Centres of Excellence Program of the Government of Canada, the Natural Sciences and Engineering Research Council, and the participation of PRECARN Associates Inc.

7 . References

[1] C. Archibald, C. Merritt, "Pose Determination of Known Objects from Sparse Range Images," *Proc. 2nd Conf. on Intelligent Autonomous Systems*, Amsterdam, The Netherlands, Dec. 1989.

[2] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. on PAMI*, Vol. PAMI-8, No. 6, Nov. 1986, pp. 679-698.

[3] W.E.L. Grimson, T. Lozano-Perez, "Localizing Overlapping Parts by Searching the Interpretation Tree," *IEEE Trans. on PAMI*, Vol. PAMI-9, No. 4, July 1987, pp. 469-482.

[4] Y. Lamdan, H.J. Wolfson,"Geometric Hashing: A General and Efficient Model-Based Recognition Scheme," *Proc. ICCV*, Tampa, Fla, Dec. 1988, pp. 238-249.

[5] H. Samet, "*The Design and Analysis of Spatial Structures: Computer Structures, Image Processing and CSG*," Addison Wesley, Reading, Mass, 1989.

[6] G. Stockman,"Object Recognition and Localization via Pose Clustering," *Computer Vision, Graphics and Image Processing*, Vol. 40, 1987, pp. 361-387.

[7] M. Soucy, D. Laurendeau, "A robust multi-operator 3-D edge detection approach for range image segmentation", in *Proc. of the Canadian Conference on Electrical and Computer Engineering*, Ottawa, Canada, Sept. 1990, pp. 37.4.1-37.4.4