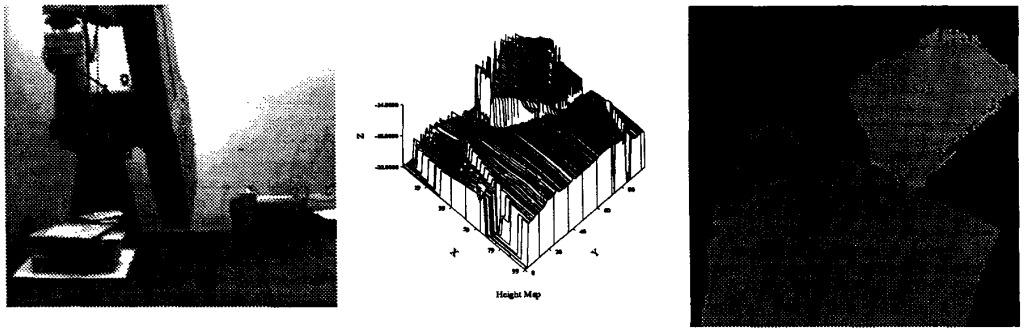


(a) (b) (c)

Figure 1: (a) A pile of postal objects, (b) range data for the scene, and (c) the segmented range map for the scene.

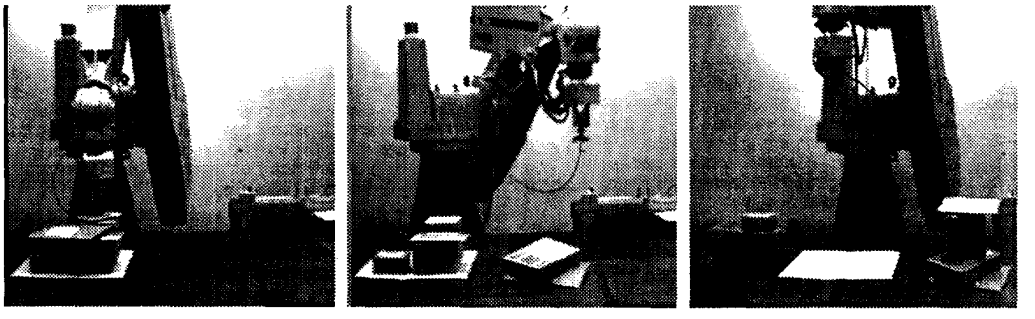


Figure 2: Seven postal objects in a pile are picked up and stacked by a robot using INGEN.

2. OVERVIEW OF INGEN

The INGEN system consists of five modules: Hypothesis Generation, Attribute Refinement, Merging, Aggregation, and Geometric Reasoning. The flow of information between these modules is shown in Fig. 3. We will first give an overview of the flow of control in INGEN and then will discuss the operation each module individually. We concentrate on the Geometric Reasoning Module and provide a brief discussion of the operation of the other modules.

The overall flow of control in INGEN is controlled by the Focus of Attention Mechanism. The first step is hypothesis generation. As described previously, an object hypothesis is created for each surface in the scene and added to the Current Scene Interpretation. As object hypotheses are refined the Current Scene Interpretation database is continuously updated. The next step is Attribute Refinement. For each hypothesis in the Current Scene Interpretation we compute a number of attributes in an entirely data-driven manner. The third step is Merging. The surfaces of all of the object hypotheses (single-surface object hypotheses at this point) are examined and if hypotheses with compatible surfaces (coplanar, cocylindrical, or coirregular) are found, and the hypotheses themselves are compatible, then they are merged into a single object hypothesis. The Attribute Refinement Module is then called upon to recompute the attributes of the newly modified hypothesis. The fourth step is Aggregation. The relations between adjacent surfaces, and therefore hypotheses, are examined and if it is determined that the adjacent hypotheses actually are parts of the same object then they are combined. The Attribute Refinement Module is then called upon to recompute the attributes of the newly modified hypothesis. Notice that merging involves combining hypotheses that are separate because surfaces were segmented into multiple parts due to noise or occlusion. Aggregation, on the other hand, combines hypotheses that are separate because distinct object surfaces were present in the scene. The final step is Geometric Reasoning. When a single range sensor is used there will always be uncertainty about the extent of objects in the direction away from the sensor. The geometric reasoning process can be thought of as "growing" the object in a particular direction until it comes into contact with another object in the scene. This allows INGEN to determine the maximum extent of the object and it also helps to ensure that the scene interpretation is geometrically consistent by ensuring that object hypotheses do not intersect one another. As before, the Attribute Refinement Module is then called upon to recompute the attributes of the newly modified hypothesis.

The most important aspect of the flow of control within INGEN is that all of the object hypotheses are created at the beginning of the recognition process and all are refined incrementally and simultaneously until the final scene interpretation is produced. This approach is necessary because the geometric reasoning process

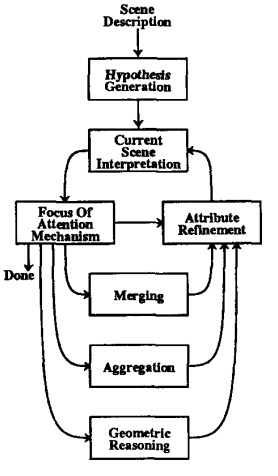


Figure 3: The INGEN system.

requires information about the entire scene interpretation to find the extent of a particular object. Although there might appear to be some redundancy in INGEN because the Attribute Refinement Module computes values for object attributes as many as four times (after hypothesis generation, merging, aggregation, and geometric reasoning) during the recognition process, this is not the case. The Attribute Refinement Module will only recompute attributes that depend on attributes that have been changed by some other module.

Currently, when INGEN makes a hypothesis combination decision it is committed to that decision even if further processing (attribute refinement or geometric reasoning) uncovers a problem. A backtracking control scheme is under development which will handle this situation. It will allow the Merging and Aggregation Modules, which are the largest sources of error in INGEN, to back out of decisions to combine hypotheses if the Attribute Refinement or Geometric Reasoning Modules detect problems with the combined hypothesis. Hypotheses will be evaluated based on a data fit measure and a visibility measure. The data fit measure will evaluate how well the hypothesis surfaces and edges fit the model surfaces and edges. The visibility measure will evaluate how well the visible parts of the object hypothesis compare with the parts of the object that are expected to be visible when occlusion has been taken into account.

In Table 1 we summarize some interesting characteristics of these modules. The Focus column shows the extent of the interpretation which the module focuses on: S refers to single objects and A refers to all objects. The Driven column shows the basic approach of the module: D refers to a data-driven approach and M refers to a model-driven approach. The Features column shows the data features used by the module: S refers to surfaces, E refers to edges, V refers to vertices, and R refers to relations.

As we discuss the operation of each module we will make reference to the two postal scenes shown in Figs. 4 and 5. This data was supplied by the GE/RCA - Advanced Technology Labs. INGEN has also been used successfully with data supplied by SRI and ERIM as well as with data from various sensors at the Purdue Robot Vision Lab. As shown in Fig. 4, scene 1 contains three objects. The topmost object is a letter which is lying on two boxes, one of which is lying on top of the other. As shown in Fig. 5, scene 2 contains four objects. The topmost object is a flat (postal terminology for large thick letter or a magazine) which lies across a box and divides it into two segments. The other two objects are a bundle of letters and a package of business reply cards.

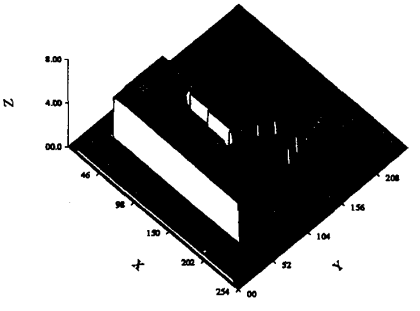
Table 1: INGEN module characteristics.

Module	Focus	Driven	Features
Hypothesis Generation	S	D	S
Attribute Refinement	S	D	SEVR
Merging	A	D	SER
Aggregation	A	D	SER
Geometric Reasoning	A	M	SEV

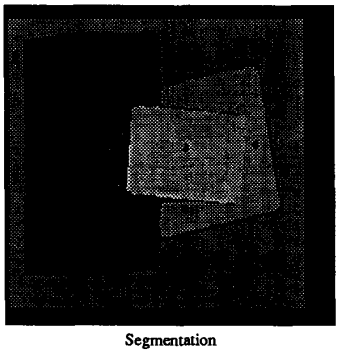
3. HYPOTHESIS GENERATION

The accuracy of the initial hypotheses used by INGEN is less critical than in most other recognition systems because instead of just verifying the correctness of the initial hypotheses INGEN will actually improve them. For each surface in the scene we create an object hypothesis. Initially, each object inherits its attributes from the attributes of its single surface. Other information available for the scene such as edges, vertices, and relations is used for refinement but is not used for hypothesis generation. Only the most basic information is required at the start of the recognition process. Later steps of the recognition process will correct initial errors and improve the accuracy of the object hypotheses.

The input to INGEN is a scene description consisting of four types of features extracted from the range data of a scene: surfaces, edges, vertices, and relations. All of the features have sets of attributes which characterize them. Surfaces include planar, cylindrical and irregular types. The irregular type is used to characterize surfaces that are generally planar but have wide variations in the local

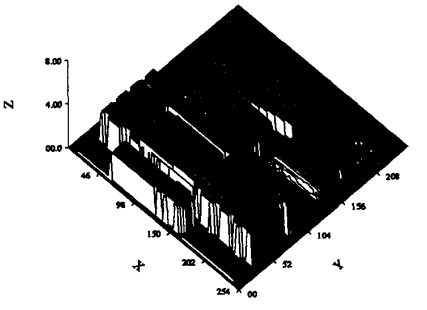


Range Plot

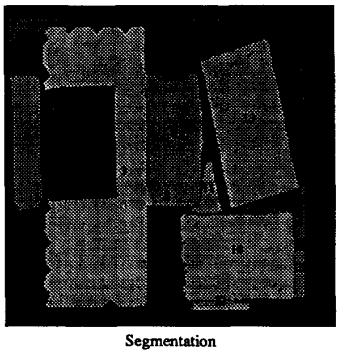


Segmentation

Figure 4: Range data plot and segmentation for scene 1.



Range Plot



Segmentation

Figure 5: Range data plot and segmentation for scene 2.

surface normals. Edges are straight line features which separate surfaces. Vertices are points where edges are joined. A relation involves two surfaces which are adjacent in the scene and any number of edges which lie along their common boundary. Some relation attributes are: boundary type (concave, convex, occluded, occluding, or unknown), local measurements of the angle and jump discontinuities at the boundary, and global measurements such as the angle between the average normals of the surfaces.

The initial object hypothesis consists mostly of information derived from the surface which supports it. The object type is based on the surface type. A cylindrical surface gives rise to a cylindrical object hypothesis. A planar surface gives rise to a parallelepiped object hypothesis. An irregular surface gives rise to an irregular object hypothesis. Irregular objects are handled the same as parallelepiped objects except that special allowances are made during the refinement process to accommodate the irregularity of their surfaces and edges. The average surface normal and the major and minor axes specify the orientation of the object. The center of mass of the surface specifies the position of the object. A homogeneous transform for the object pose is computed from the axes and the position. The extents along the three axes define the size of the object. The values originally assigned to these attributes will most likely be changed as better estimates are available and values for other attributes that were not included in the initial hypothesis will be computed during attribute refinement processing.

The initial object hypotheses can be inaccurate in a number of ways, all of which can be corrected by the hypothesis refinement process. For example, the position, orientation, or dimensions of the object may be incorrect but will be recomputed during the attribute refinement and geometric reasoning processes. An object with multiple surfaces visible in the scene will cause multiple object hypotheses to be formed. During the merging or aggergation processes, these hypotheses will be combined into a single object hypothesis.

4. ATTRIBUTE REFINEMENT

The task of the Attribute Refinement Module is to compute values for object attributes. The processing is essentially data-driven except that some procedures use different techniques for the three primary types of objects: parallelepipeds, cylinders, and irregulars.

Currently, there are fourteen refinement procedures which carry out the processing of the Attribute Refinement Module. Each refinement procedure has a set of object attributes as its input and produces a set of attributes as its output. An attribute may be used as input to several procedures and may also be produced as output by several procedures. This arrangement allows for the bootstrapping of attributes. One procedure can find a rough estimate for a particular attribute and another procedure can then use this estimate along with other attributes to determine a superior estimate.

The Attribute Refinement Module is called upon many times during the recognition process. It is called for the first time immediately after hypothesis generation. In this step, known as the Initial Attribute Refinement step, every object hypothesis has its attributes computed by every refinement procedure. Subsequently, the calling of the Attribute Refinement Module is known as Supplementary Attribute Refinement and is triggered when another module changes an attribute of an object hypothesis. For this case, the Attribute Refinement Module is given an object and a list of attributes that have been changed for that object and it calls all of the refinement procedures that depend on those attributes. Forward chaining is used to determine which other procedures depend (directly or indirectly) on those attributes and therefore need to be called to propagate the change.

Here we offer a brief description of some of the more important refinement procedures. The *surface confidence* refinement procedure produces a measure of the confidence of the classification of the surface types for the object's constituent surfaces. The *general object type* refinement procedure examines the types of the surfaces and relations of the object hypothesis and its dimensions in order to determine the general object type: parallelepiped, cylinder, irregular, prism, or other. The *align* refinement procedure looks at the edges of the object and defines new object axes if certain conditions are satisfied. The procedure looks for object edges that are either occluding or convex type. Using only these edge types ensures that the system is not fooled by shadows or occlusion effects. If the longest of these edges has a length that is a significant fraction of the initially computed length of the object, then the direction of the major axis is changed to the direction of this edge and the direction of the minor axis is changed to be perpendicular to the major axis and the normal axis. If no appropriate edges are found then the initially computed moment-based major and minor axes are retained. In reference to example scene 1, the major axes for the leftmost box and the letter are changed only slightly by the refinement process because the letter is unoccluded and the box is only slightly occluded. The moment-based method provides good estimates of the axes of objects that are unoccluded but can be very inaccurate for occluded objects. The rightmost box in example scene 1 demonstrates this problem. Three complete edges of the object are visible but due to occlusion only about half of area of the object is visible. The edge-based method produces a much better estimate of the major axis in this case. The *extent* refinement procedure looks at all of the object vertices and finds their distances from the object position. The maximum distances in both directions along each axis are used to define the dimensions of the object. The position is also redefined to be centered on the major and minor axes and to lie on the primary surface of the object. Extent refinement is necessary whenever axis refinement adjusts the orientation of the object. The *height to table* refinement procedure defines the object height to be the vertical component of the object position. This gives exact results for an object lying flat on the table. For objects that are tilted or lying on other objects the Geometric Reasoning Module will produce better estimates. Thus, in example scene 1 the two boxes have their heights estimated accurately but the letter height is overestimated. The *height combine* refinement procedure produces a value for the object height attribute based on the object type, the minimum and maximum height estimates from the extent attributes, the height above the table, and values produced by the Geometric Reasoning Module. The *homogeneous transformation* refinement procedure uses the object axes and the object position to define the homogeneous transformation for the object. This transformation is suitable for use by a robot with a suction gripper for picking up the object. It is also used by the bounding box fitting procedure to find the model-to-scene and scene-to-model transformations. The *area* refinement procedure finds the visible area of the object and the area of the object that should be visible. These values can be used to determine the degree to which the object is occluded. The *confidence* refinement procedure produces a measure of the confidence in the object hypothesis. The *postal type* refinement procedure determines the appropriate postal category for the object based on size and shape information. The *bounding box* refinement procedure fits a bounding box around the object hypothesis and determines the model-to-scene and scene-to-model transformations for the object which are used by the Geometric Reasoning Module.

Figs. 6 and 7 show the results for the two example scenes after the hypothesized objects have been through Initial Attribute Refinement. Each figure shows four views of the scene in wireframe form. Included are orthographic projections for the top (x-y plane), front (y-z plane), and side (x-z plane) views and an axonometric projection. In the first scene, note that the letter is hypothesized as a box with its estimated height being its height above the table. This results in the letter intersecting both of the other objects and the table. In the second scene note that the leftmost box has been hypothesized as two objects, the flat lying on that box has also been hypothesized as two objects, and the table has been hypothesized as six objects.

5. MERGING

In cluttered scenes object surfaces will frequently be separated into multiple adjacent surface segments due to segmentation errors or into non-adjacent surface segments because of occlusion. By merging object hypotheses based on compatibility and continuity criteria the Merging Module can solve these problems.

Merging takes place in two phases. The first phase, merging based on compatibility, uses information derived from the relations between adjacent surfaces (and objects) to merge them. The second phase, merging based on alignment, uses information about surface and object characteristics to merge surfaces which are not adjacent. The use of object attributes makes it necessary to do the merging after object hypotheses have been formed. Attempts to carry out merging without object information led to problems with merging distinct objects and not merging parts of objects. The primary reason

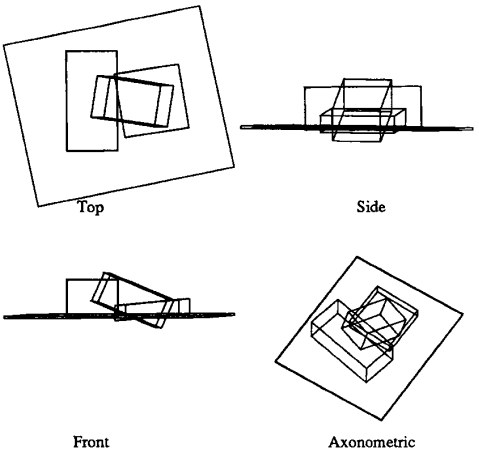


Figure 6: Hypothesized objects for scene 1.

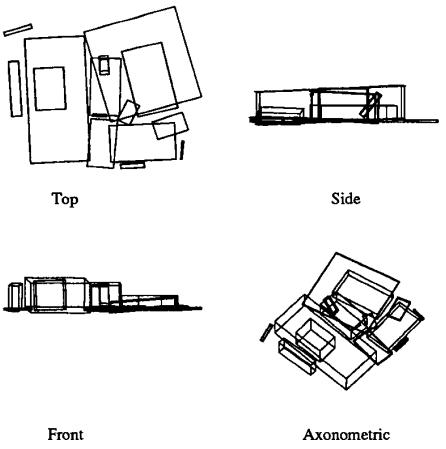


Figure 7: Hypothesized objects for scene 2.

for using object information is the object major axes provided by the refinement procedures are superior to the moment based axes for each individual surface.

Our first example scene did not require any merging. In the second example scene merging was critical to finding the correct scene interpretation. The large box was divided into two widely separated segments because it was occluded by the flat. Also note that the flat itself was segmented into two adjacent segments because of dropouts in the range data caused by the black outline of the mailing label. In both cases the two surfaces are merged together into a single object hypothesis.

6. AGGREGATION

The Aggregation Module combines pairs of object hypotheses that correspond to different surfaces of the same object into single hypotheses. The difference between this module and the Merging Module is that in this module we are not looking for parts of the same surface in two hypotheses. Instead, we are looking for distinct surfaces of a single object that have been hypothesized as separate objects.

Because we have only a few object types and assume that all objects are convex we can make the assumption that two surfaces that are convexly adjacent are likely to be surfaces of the same object. This approach also has the advantage that is can handle objects that we don't have models for. Any group of convexly related surfaces can be considered to be an object of type other.

7. GEOMETRIC REASONING

The task of the Geometric Reasoning Module is to determine size parameters for object hypotheses that can't be determined from the scene data. It resolves uncertainty in one of the dimensions of an object by "growing" it until it contacts another object in the scene. The point of contact need not be visible in the scene data. This process also helps to ensure that the scene interpretation is geometrically consistent by ensuring that object hypotheses do not intersect each other.

The geometric reasoning algorithm used in INGEN is based on a single operation: finding the maximal extent of an object hypothesis in a particular direction. Before we describe our algorithm we must introduce some terminology that will help to clarify the discussion. The *object* is the object hypothesis for which we wish to determine the maximum value of a particular size parameter. The *obstacles* are all of the other object hypotheses in the scene which potentially physically constrain the object. We use this terminology because it is convenient to think of the computation of the maximal extent of an object as the "growing" of the object until it comes in contact with an obstacle.

The algorithm consists of four steps.

1. Find the scene-to-model transformation for the object. We assume, without loss of generality, that the parameter to be found is the object's extent along the positive z axis.
2. Use the scene-to-model transformation to transform all obstacles into the model coordinate frame of the object.
3. Find the maximum extent of the object along the positive z axis such that the object is in contact with at least one obstacle in the scene and its volume does not intersect the volume of any obstacle in the scene.
4. Use the model-to-scene transformation to transform the extent measurement back into the scene coordinate frame and thus find its actual value.

7.1. Step 1

The determination of the scene-to-model transformation in step one is carried out during the attribute refinement stage of processing by the bounding box refinement procedure. INGEN uses bounding boxes to model objects as parallelepipeds but this algorithm will work with more complex object models.

For the typical recognition problem, the relationship between a point, $\mathbf{p}_M$, in the model coordinate frame and a point, $\mathbf{p}_S$, in the scene coordinate frame is defined by $\mathbf{p}_S = \mathbf{T}\mathbf{p}_M$ where $\mathbf{T}$ is a homogeneous transformation matrix for the object pose (position and orientation). On the other hand, in INGEN, the object models do not contain metrical information so a pose transformation alone does not completely specify the relationship between the object and the model. All objects are modeled by a parallelepiped model which is defined by their bounding box. The parallelepiped model is a unit cube which requires three scale factors as well as a pose transformation in order to be matched with objects in the scene. The scale factors define the length of the objects along the three perpendicular axes in the model coordinate frame. When the scale factors are incorporated into the transformation between the scene and model coordinate frames we have $\mathbf{p}_S = \mathbf{T}\mathbf{S}\mathbf{p}_M$ where the scale transformation, $\mathbf{S}$, is defined by:

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus, the transformations between the model and scene coordinate frames are defined by:

$$\mathbf{p}_s = \mathbf{T}\mathbf{S}\mathbf{p}_m = \mathbf{T}_{m2s}\mathbf{p}_m$$
$$\mathbf{p}_m = \mathbf{S}^{-1}\mathbf{T}^{-1}\mathbf{p}_s = \mathbf{T}_{s2m}\mathbf{p}_s$$

It is important to note that in the first equation the scaling transformation is pre-multiplied by the pose transformation so that scaling takes place in the model coordinate frame. Likewise with the second equation where the inverse scaling transformation is post-multiplied by the inverse pose transformation.

7.2. Step 2

In step two all of the objects are transformed into the model coordinate frame of the object of interest. Fig. 8 shows the hypothesized objects of scene 1 after being transformed into the model coordinate frame of the letter. The letter is transformed into a unit cube, which is upside down with respect to Fig. 6, and the other objects are transformed accordingly. i.e. The visible top surface of the letter becomes the bottom surface of the unit cube in the model coordinate frame and the other objects appear above the letter.

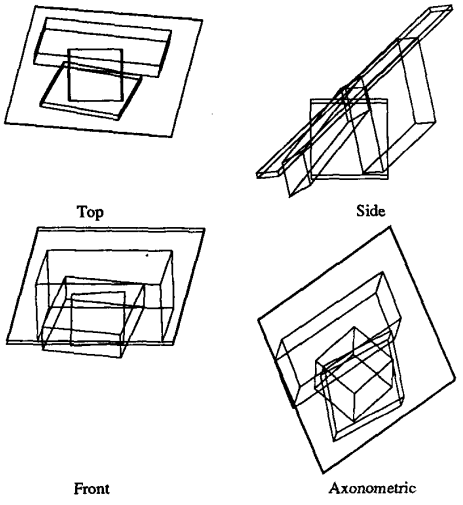


Figure 8: Hypothesized objects for scene 1 after being transformed into the model coordinate frame of the topmost object.

7.3. Step 3

Clearly, step three of this algorithm is the most difficult one. However, the transformation of the obstacles into the model coordinate frame of the object which was carried out in step two simplifies the computations in step three. Our approach borrows some computational techniques from ray casting algorithms in the computer graphics field and from path planning algorithms in the robotics field. The ray casting technique that we borrow is to transform our object and obstacles into a coordinate frame where the object hypothesis is a simple primitive object of fixed size and shape. The path planning technique that we borrow is the use of both parametric and implicit representations of surfaces and edges to aid in the computation of intersections. Roth [24] discusses ray casting techniques and algorithms and Lozano-Pérez [20] discusses the configuration space approach to robot motion planning.

Three types of contact are possible between polyhedral objects and polyhedral obstacles. We use the traditional convention [20] to name these interactions: *Type A* - an object surface contacting an obstacle vertex, *Type B* - an object vertex contacting an obstacle surface, and *Type C* - an object edge contacting an obstacle edge.

The three types of contact are shown in Fig. 9. Note that the objects and the obstacles are positioned in the object's object-centered model coordinate frame. Thus, the obstacles which are below the object in the scene appear above the object in the model coordinate frame.

For each object-obstacle interaction we need to check for all three types of contact. The maximum object height (in the model coordinate frame) is determined by the minimum height contact with the obstacle. This process is carried out for each obstacle in the scene and the results are then combined to find the maximum height for the object which is determined by the minimum height contact with any obstacle.

In the following sections we describe the computational techniques for dealing with the three types of contact. It is important to note that all of the computations are carried out in the model coordinate frame of the object of interest. We will always be finding the maximum extent of the object along the positive z axis, which we refer to as the height of the object, in the model coordinate frame. Thus, when we refer to the top surface of the object we are referring to the surface which has a surface normal that points in the positive z direction in the model coordinate frame regardless of it's actual location and orientation in the scene.

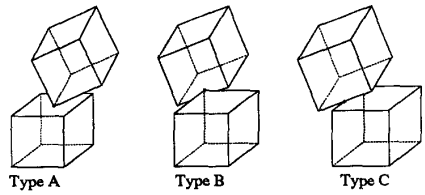


Figure 9: Type A, B, and C contacts.

7.3.1. Type A - An object surface contacting an obstacle vertex

For type A contact we need to find the obstacle vertex which will contact the top surface of the object at the lowest point. The procedure is to find all obstacle vertices that could potentially contact the top surface of the object and then find the one with the smallest z component. For parallelepiped obstacles the only vertex that can contact the topmost surface of the object is the one with the smallest z component. For all of the other vertices an edge will necessarily contact the object at a point that is lower than the vertex so these vertices can be ignored. Thus, only one of the the eight vertices for each parallelepiped obstacle is tested for constraining the object.

For a parallelepiped object the scene-to-model coordinate transformation results in the object being a unit cube in the positive octant with a vertex at the origin. Only obstacle vertices which satisfy $0 \le x \le 1$ and $0 \le y \le 1$ can potentially contact the top surface of the parallelepiped.

7.3.2. Type B - An object vertex contacting an obstacle surface

For parallelepiped objects we need to find the obstacle surface which will contact an object vertex at the lowest point. The procedure is to find the contact points for all obstacle surfaces that could potentially contact the top vertices of the object and then find the one with the smallest z component. Potential contact points are the intersection points of the four lines defined by the vertical edges (parallel to the z axis) of the parallelepiped object and the obstacle surfaces. The four lines are defined by: $\{ x = 0, y = 0 \}$, $\{ x = 0, y = 1 \}$, $\{ x = 1, y = 0 \}$, and $\{ x = 1, y = 1 \}$. We only need consider obstacle surfaces that have their average surface normal pointing in the negative z direction. For all of the other surfaces some other part of the obstacle will necessarily contact the object at a point that is lower than the surface so these surfaces can be ignored. Thus, only three of the six surfaces of each parallelepiped obstacle are tested for constraining the object.

All of the planar obstacle faces belong to rectangular parallelepipeds so the faces will be parallelograms when transformed into the model frame of the object. By using the parametric representation for these faces the intersection problem can be solved efficiently. We select one vertex, (x_0, y_0, z_0), of the face as its origin and use the two adjacent vertices, (x_1, y_1, z_1) and (x_2, y_2, z_2), to define oblique axes for the parameterization of the surface. The numbering of the adjacent vertices should be defined so that the normal of the face points away from the object. Using u and v as parameters the parametric equations for the parallelogram face are:

$$x = x_0 + uf_1 + vf_2, \quad y = y_0 + ug_1 + vg_2, \quad z = z_0 + uh_1 + vh_2$$

where:

$$f_1 = x_1 - x_0, \quad g_1 = y_1 - y_0, \quad h_1 = z_1 - z_0$$
$$f_2 = x_2 - x_0, \quad g_2 = y_2 - y_0, \quad h_2 = z_2 - z_0$$

By substituting the x and y values from the line equation pairs into the x and y parametric equations for the plane we can solve for the two surface parameters u and v:

$$u = \frac{g_2(x - x_0) - f_2(y - y_0)}{f_1 g_2 - f_2 g_1}, \quad v = \frac{-g_1(x - x_0) + f_1(y - y_0)}{f_1 g_2 - f_2 g_1}$$

If $0 \le u \le 1$ and $0 \le v \le 1$ then the intersection point lies within the obstacle face. We then substitute u and v into the z parametric equation to find the z coordinate of the intersection point.

7.3.3. Type C - An object edge contacting an obstacle edge

For type C contact we need to find the obstacle edge which will contact the top edge of the object at the lowest point. The procedure is to find all points at which obstacle edges intersect the vertical surfaces of the object and then find the one with the smallest z component. We can exclude some edges from consideration because of the surfaces that they bound. All of the edges of the surface with a surface normal that forms the most positive dot product with the positive z axis are excluded. Also, the edge between the other two surfaces with surface normals that form positive dot products with the positive z axis are excluded. Thus, only seven of the twelve edges of each parallelepiped obstacle are tested for constraining the object.

For parallelepiped objects the contact points are the intersection points between obstacle edges and the four vertical faces (parallel to the z axis) defined by the four plane equations: $x = 0$, $x = 1$, $y = 0$, and $y = 1$. For planes $x = 0$ and $x = 1$ we are interested in intersection points such that $0 \le y \le 1$. For planes $y = 0$ and $y = 1$ we are interested in intersection points such that $0 \le x \le 1$.

By using the parametric representation for obstacle edges we can simplify the intersection computations. Edges are defined by their endpoint vertices: (x_0, y_0, z_0) and (x_1, y_1, z_1). Using t as the parameter, the parametric equations for the edge are:

$$x = x_0 + tf, \quad y = y_0 + tg, \quad z = z_0 + th$$

where:

$$f = x_1 - x_0, \quad g = y_1 - y_0, \quad h = z_1 - z_0$$

By substituting the x or y value from the plane equation into the appropriate edge equation we can solve for the edge parameter t:

$$t = \frac{x - x_0}{f}, \quad t = \frac{y - y_0}{g}, \quad t = \frac{z - z_0}{h}$$

If $0 \le t \le 1$ then the intersection point lies within the obstacle edge. We then substitute t into the other parametric equations to find the other coordinates of the intersection point.

7.4. Step 4

The result from step three is a value for the maximum extent of the object along the positive z axis in the current model coordinate frame of the object. Note that this coordinate frame depends on the previous dimensions of the object. A maximum extent value less than one indicates that the old dimension was too large and a value greater than one indicates that the old dimension was too small. The new object dimension is obtained in step four by simply multiplying the value found in step three by the previous value for the object dimension.

Figs. 10 and 11 show the final results produced by INGEN for the two example scenes. Notice that in the first scene the letter has been recognized correctly. Also notice that one of the boxes extends below the other one so it appears to be floating in air. There is no way for the system to know how far the box extends under the other one and we don't test for or require physical stability. One approach that would help to solve this problem would be to extend the box under the other one until it contacts known empty space. The same basic approach would be used but the representation of empty space would necessarily be more complex than the parallelepipeds that we currently deal with. In the second scene the object that is resting on top of the box has been correctly identified as a flat because of geometric reasoning.

While we currently use only bounding boxes the approach described here can be extended to handle more complex objects in two ways. The first is to define other types of primitive objects besides the parallelepiped bounding box. Spheres, trapezoids, tori, generalized cylinders, and polyhedra are candidates that could be considered. However, with the possible exception of polyhedra and cylinders, they will probably be of limited usefulness in most cases. This approach requires the development of new intersection algorithms for each new type of object. The second path is to consider composite objects which are constructed from combinations of the primitive objects. For the simplest case, primitives are combined by the attachment operation. In this case, the techniques that we are currently using are directly applicable. All that is required is that we consider each of the primitives individually (even if they are part of a more complex object) and then combine the results as if each primitive was a separate obstacle. The only changes necessary would be the overhead required to group the primitives together as objects. Objects would be defined by a single object coordinate frame and a set of primitives with transformations relating them to the object coordinate frame. The more complex, and more general, approach is to use Constructive Solid Geometry (CSG) to define objects. The CSG representation represents objects as sets of primitives combined by regularized volumetric union, intersection, and subtraction operations. As with composite objects, primitives can be considered individually and the results combined for each object. However, the union, intersection, and subtraction operations make the combination of results much more complex. Ray casting programs use this approach successfully [24] for finding the intersections of lines with CSG objects. Our problem of finding the intersections between CSG objects is not quite as simple but is computationally feasible.

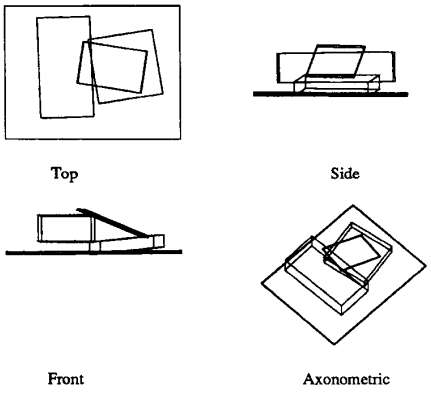


Figure 10: Final results for scene 1.

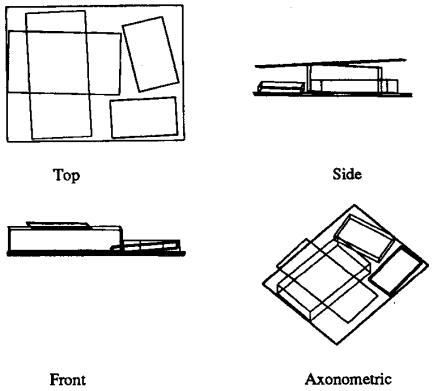


Figure 11: Final results for scene 2.

8. CONCLUSION

We have discussed the INGEN system which addresses the problem of generic object recognition in the postal domain. There are three important aspects of this system which make it particularly useful for generic object recognition. Data-driven processing reduces its sensitivity to surface and edge irregularities. The hypothesize-and-refine control structure allows varied approaches to computing object attributes and incremental updating of the scene interpretation. The use of information derived from geometrical contacts between objects enables it to determine object dimensions.

9. ACKNOWLEDGMENTS

We would like to thank Dr. Frederick R. Glickman and Mr. Gary P. Herring of the US Postal Service Office of Advanced Technology for many insightful discussions as the generic object recognition project has progressed from the initial conceptualization through the current implementation. Dr. John Tan, Dr. Gerardo Garcia, Mr. Jeff Shaevel, and Dr. Marc Bastuscheck of Arthur D. Little, Inc. also deserve thanks for their advice throughout this project.

We would also like to thank Kirk D. Smith and Robert L. Cromwell for the development of the low and mid level range data processing, segmentation, and characterization system which provides the input to INGEN.

10. REFERENCES

[1] R. Bajcsy and F. Solina, "Three Dimensional Object Representation Revisited," *Proceedings of the First International Conference on Computer Vision*, IEEE Computer Society, 1987.

[2] R. C. Bolles and P. Horaud, "Configuration Understanding in Range Data," *Robotics Research: The Second International Symposium*, H. Hanafusa and H. Inoue, Eds., MIT Press, 1985.

[3] R. C. Bolles and P. Horaud, "3DPO: A Three-Dimensional Part Orientation System," *The International Journal of Robotics Research*, Vol. 5, No. 3, Fall 1986.

[4] R. A. Brooks, "Symbolic Reasoning Among 3-D Models and 2-D Images," *Artificial Intelligence*, Vol. 17, 1981.

[5] R. A. Brooks, "Model-Based Three-Dimensional Interpretations of Two-Dimensional Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 2, March 1983.

[6] C. H. Chen and A. C. Kak, "3D-POLY: A Robot Vision System for Recognizing Objects in Occluded Environments," Purdue University Technical Report TR-EE 88-48, December 1988.

[7] C. H. Chen and A. C. Kak, "A Robot Vision System for Recognizing 3-D Objects in Low-Order Polynomial Time," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 19, No. 6, 1989.

[8] C. K. Cowan, D. M. Chelberg, and H. S. Lim, "ACRONYM Model Based Vision in the Intelligent Task Automation Project," *Proceedings of the First Conference on Artificial Intelligence Applications*, IEEE Computer Society, 1984.

[9] R. B. Fisher, "Using Surfaces and Object Models to Recognize Partially Obscured Objects," *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, 1983.

[10] R. B. Fisher, "SMS: A Suggestive Modelling System for Object Recognition," *Image and Vision Computing*, Vol. 5, No. 2, May 1987.

[11] R. B. Fisher and M. J. L. Orr, "Solving Geometric Constraints in a Parallel Network," *Image and Vision Computing*, Vol. 6, No. 2, May 1988.

[12] W. E. L. Grimson, "The Combinatorics of Local Constraints in Model-Based Recognition and Localization from Sparse Data," *Journal of the Association for Computing Machinery*, Vol. 33, No. 4, October 1986.

[13] W. E. L. Grimson, "Recognition of Object Families Using Parameterized Models," *Proceedings of the First International Conference on Computer Vision*, IEEE Computer Society, 1987.

[14] W. E. L. Grimson and T. Lozano-Pérez, "Model-Based Recognition and Localization from Sparse Range or Tactile Data," *The International Journal of Robotics Research*, Vol. 3, No. 3, Fall 1984.

[15] W. E. L. Grimson and T. Lozano-Pérez, "Localizing Overlapping Parts by Searching the Interpretation Tree," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 4, July 1987.

[16] A. C. Kak, K. D. Smith, A. J. Vayda, and R. L. Cromwell, "Range Image Interpretation for Postal Object Recognition," *Third United States Postal Service Advanced Technology Conference*, 1988.

[17] A. C. Kak, A. J. Vayda, R. L. Cromwell, W. Y. Kim, and C. H. Chen, "Knowledge-Based Robotics," *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, IEEE Computer Society, 1987.

[18] A. C. Kak, A. J. Vayda, R. L. Cromwell, W. Y. Kim, and C. H. Chen, "Knowledge-Based Robotics," *International Journal of Production Research*, Vol. 26, No. 5, 1988.

[19] A. C. Kak, A. J. Vayda, K. D. Smith, and C. H. Chen, "Recognition Strategies for 3-D Objects in Occluded Environments," *Traditional and Non-Traditional Robotic Sensors*, T. C. Henderson, Ed., NATO Advanced Research Workshop Series F, Springer-Verlag, Berlin, 1990.

[20] T. Lozano-Pérez, "A Simple Motion-Planning Algorithm for General Robot Manipulators," *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 3, June 1987.

[22] P. G. Mulgaonkar, C. K. Cowan, and J. DeCurtins, "Scene Description Using Range Data," *Proceedings of the Workshop on the Interpretation of 3D Scenes*, IEEE Computer Society, 1989.

[21] P. G. Mulgaonkar and J. DeCurtins, "Scene Description for Object Manipulation in Unstructured Environments," *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, IEEE Computer Society, 1989.

[23] M. J. L. Orr and R. B. Fisher, "Geometric Reasoning for Computer Vision," *Image and Vision Computing*, Vol. 5, No. 3, August 1987.

[24] S. C. Roth, "Ray Casting for Modeling Solids," *Computer Graphics and Image Processing*, 18, 1982.

[25] F. Solina and R. Bajcsy, "Shape Recovery of Mail Pieces Using Deformable Models," *Third United States Postal Service Advanced Technology Conference*, 1988.

[26] E. L. Walker, M. Herman, and T. Kanade, "A Framework for Representing and Reasoning about Three-Dimensional Objects for Vision," *AI Magazine*, Vol. 9, No. 2, Summer 1988.