CAMERA PLACEMENT PLANNING AVOIDING OCCLUSION: TEST RESULTS USING A ROBOTIC HAND/EYE SYSTEM

Kostantinos Tarabanis

Department of Computer Science
Columbia University

Roger Y. Tsai

IBM T.J. Watson Research Center
Manufacturing Research
Yorktown Heights, New York 10598

Technical Report No. CUCS 501-89

Abstract

Camera placement experiments are presented that demonstrate the effectiveness of a viewpoint planning algorithm that avoids occlusion of a visual target. A CCD camera mounted on a robot in a hand-eye configuration is placed at planned unobstructed viewpoints to observe a target on a real object. The validity of the method is tested by placing the camera inside the viewing region, that is constructed using the proposed new sensor placement planning algorithm, and observing whether the target is truly visible. The accuracy of the boundary of the constructed viewing region is tested by placing the camera at the critical locations of the viewing region boundary and confirming that the target is barely visible. The corresponding scenes from the candidate viewpoints are shown demonstrating that occlusions are properly avoided.

This work was supported in part by DARPA contract N00039-84-C-0165 and IBM T.J. Watson Research Center, Manufacturing Research.

Camera Placement Planning Avoiding Occlusion: Test Results using a Robotic Hand/Eye System

Kostantinos Tarabanis[1]
Computer Science Department
Columbia University
New York, NY 10027

Roger Y. Tsai
Manufacturing Research
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

ABSTRACT

Camera placement experiments are presented that demonstrate the effectiveness of a viewpoint planning algorithm that avoids occlusion of a visual target. A CCD camera mounted on a robot in a hand-eye configuration is placed at planned unobstructed viewpoints to observe a target on a real object. The validity of the method is tested by placing the camera inside the viewing region, that is constructed using the proposed new sensor placement planning algorithm, and observing whether the target is truly visible. The accuracy of the boundary of the constructed viewing region is tested by placing the camera at the critical locations of the viewing region boundary and confirming that the target is barely visible. The corresponding scenes from the candidate viewpoints are shown demonstrating that occlusions are properly avoided.

Introduction

One of the major factors contributing to the development cost and time for machine vision applications is the determination of the placement of the camera and the associated optical setup. Being able to automatically determine the sensor placement is important for reducing the development cycle and cost in today's manufacturing environment. Furthermore, with the increasing emphasis on process control, it has become increasingly important to have a machine vision system that automatically adapts itself to the changing process requirements, which are updated frequently during process optimization. The ability to automatically determine sensor placement given any process requirement is therefore desirable.

The geometric and physical information that is available today in manufacturing in the form of CAD models of parts can be used for this purpose of automatically generating sensor placement strategies. Currently, CAD/CAM models are used only in the design and manufacturing of objects. However, this knowledge can be used in other important tasks at later stages of the manufacturing process as well (Ref. 15). For example, the information regarding the geometry of a part as well as its material properties (e.g. color, surface finish), that are typically contained in the CAD/CAM model, can be used to automatically generate sensor and illuminator placement strategies for the inspection and gaging of this part. In general, this will lead to a vertically integrated manufacturing environment that would also operate more flexibly and more autonomously as various processes can be planned and be performed automatically.

In this paper, occlusion avoiding camera placement results are presented for a robotic hand-eye system. First, an outline is given of the sensor placement planning algorithm that avoids optical occlusion given any specified polygonal target and occluding polyhedral solid opaque object. The details of this algorithm can be found in Ref. 3. Results are then presented for

[1] this author was supported in part by DARPA contract N00039-84-C-0165 and in part by Manufacturing Research, IBM T.J. Watson Research Center.

actual camera placement in a robotic workcell at poses where a given visual target can be viewed in its entirety, demonstrating the effectiveness of the algorithm.

Outline of the Visibility Planning Algorithm

The algorithm generates regions in three-dimensional space from where a given visual target can be viewed totally without being obstructed by occluding polyhedral objects. Existing approaches to this problem can be found in Ref. 1 and Ref. 2.. The first technique deals primarily with convex occluding objects, while the latter incorporates a generate-and-test strategy for viewpoints on a tesellated viewing spherical surface alone.

Our visibility planning algorithm first considers the faces of the occluding polyhedron as occluding polygons. It then decomposes the problem of determining occlusion-free regions in space for general occluding and target polygons into similar subtasks between convex polygons. The two decompositions introduced for this purpose, along with the method to solve the convex subproblems, are described in the next section. The occluded regions of the faces of the polyhedron are then unioned to generate the occluded region of the polyhedron as a whole.

Material-Hole Decomposition (Loop Decomposition)

Figure 1 shows an occluding polygon with a hole in it. The occluded region for viewing a target T with an occluding polygon B containing a hole H in it is equal to the occluded region caused by B (shown in Figure 2) less the region in space where the camera can view the target T through the hole H (shown in Figure 3). The first subtask is to determine the occluded region caused by the *material* polygon B. The second subtask is to determine the viewing region through the hole H. The resultant occluded region is equal to the difference of the above two regions, shown in Figure 4. In general, the following formula holds:

$$Occluded\ Region_{resultant} = Occluded\ Region_{material} - ViewingRegion_{hole\ within\ material} \qquad (1)$$

However, two levels of decomposition are actually needed to accomplish the task in general. If the occluding polygon B or the hole H within B or the target T are not convex, then the two subtasks are still not convex, and therefore, another level of decomposition is needed, as shown in the next section.

Polygons in general may contain several loops (cycles of edges) inside their outermost boundary. These loops can also be nested. The loop decomposition partitions the original polygon into the loops that it contains and builds a tree, the *loop tree*, that represents their nesting explicitly. The interior of loops at odd levels of the tree are material regions, while loops at even levels are holes. We name the former, *material loops* and the latter, *hole loops* . In the context of the visibility computations, material and hole loops are very different. Material loops generate occluded regions in viewing space while hole loops create visible regions. The occluded region of the original polygon is computed by combining the occluded and visible regions of each of its component loops as will be explained in "The Visiblity Planning Algorithm" .

Convex Material-Gulf Decomposition (Convex Decomposition)

Consider the situation in Figure 1 discussed earlier. Suppose that the occluding polygon B is not convex but is shaped like that in Figure 5. Then it is necessary to decompose B into its convex hull B_{hull} and a gulf B_{gulf} such that

$$Polygon = Convex\ Hull - Gulfs \quad or \quad B = B_{hull} - B_{gulf} \qquad (2)$$

The resultant occluded region is equal to the difference of the occluded region of the convex hull less the viewing region of the gulf. However, to obtain the correct viewing region, the gulf should be enlarged to become B_{gulf_hole}, a hole that is equivalent to the gulf in this viewing relation as shown in Figure 6. The equivalent hole is bounded by limiting viewing rays defined by pairs of vertices, one from the target and one from the gulf. The algorithm for the construction of the equivalent hole can be found in Ref. 3. The equivalent hole shown in Figure 6 is concave and must be decomposed itself into convex parts in a similar fashion. The resultant viewing region through the gulf is shown in Figure 7, and the resultant occluded region is shown in Figure 8.

In general, the convex decomposition algorithm approximates a simple polygon (a polygon with a single loop) by a sum of convex polygons. These convex polygons can be added and subtracted in an alternating sequence to construct the original polygon. At each level of the convex decomposition algorithm, the following sequence of operations is performed:

1. the convex hull of a polygon is computed and stored.
2. the polygon is subtracted from its convex hull.
3. any convex polygons that result from the subtraction in 2 are stored.
4. any concave polygons that result from the subtraction in 2 are decomposed recursively.

The result of this decomposition is a set of convex polygons that can be arranged in a tree, which we call the *convex tree*. The original concave polygon can be generated by subtracting the convex polygons corresponding to children nodes from the convex polygon of their parent node, in a bottom-up fashion. In this way, convex polygons at odd levels of the tree are added to the sum that generates the original concave polygon (*material polygons*), while convex polygons at even levels of the convex tree (*gulf polygons*) are subtracted from this sum. In the context of the visibility computations, the occluded region of the original polygon is computed by combining the occluded and visible regions of each of its component convex polygons as will be explained in "The Visiblity Planning Algorithm" .

Convex Visibility Planning

After the above two decompositions (loop and convex), what remains is the basic convex visibility planning task. There are two kinds of convex tasks,

- Occluded region computation,

  where a convex polygon occludes a convex target, and

- Viewing region computation,

  where a convex target can be viewed through a convex hole.

The occluded region is bounded by a family of limiting separating planes defined by edge-vertex or edge-edge pairs from the occluding and target polygons such that the entire target and occluding polygons are placed in different half-spaces (Figure 2). On the other hand, the viewing region is bounded by a family of limiting separating planes defined by an edge on the hole polygon and a vertex or edge on the target polygon such that the entire target and hole polygons are placed in the same half-space (Figure 3). Both regions can be computed efficiently (Ref. 3) and then be combined to construct the occluded region of the general occluding polygon and target.

The Visiblity Planning Algorithm

The global algorithm decomposes the general visibility planning problem for an occluding polygon and target into a tree structure. This tree is a structure with each node being itself

a tree. The global tree represents the loop decomposition, while the tree at each node represents the convex decomposition. In other words, the global tree is the *loop tree* and the local trees are the *convex trees*. This tree structure is a decomposition of both the global task as well as the occluding polygon. For both trees, the odd levels are the materials and the even levels are the holes. The basic convex visibility planning subtask is applied to each node of the convex tree. Then, the occluded or viewing region for each node of the loop tree is computed by recursively subtracting the child from the parent, starting from the leaf nodes. Finally, the same recursive operation is applied to the loop tree, resulting in the global occluded region.

This tree structure provides a good mechanism for pruning for the sake of speed. Since the nodes close to the bottom of the tree (especially the convex tree) may represent fine details both for the final viewing region and for the occluding object, pruning the tree by eliminating nodes close to the bottom can be a natural "filtering" process both for the task and for the final viewing region. This is useful since in computing the viewing region, it is not necessary to determine the boundary of the viewing region precisely, although it should be *conservative* , in the sense that the viewing region can be smaller than the true viewing region, as long as all points inside the viewing region satisfy the visibility constraint. Therefore, the pruning must start from the level of the tree that represents holes (holes increase the viewing region while materials do the opposite). For objects with minute details, pruning is quite important to make the computation feasible.

Test Results

A working system for visibility planning has been implemented. In this section, we seek to demonstrate that the results produced by the working system, which incorporates the new method, are correct. This is done by performing both simulation and real experiments.

The occluding object and the target are the same for both types of experiments and are shown as CAD models in Figure 9. The target to be viewed is part of the occluding object itself, namely the top face T of the enclosed cube. Figure 13 shows the actual object and target used in the real camera placement experiments.

Simulation Experiments

In the following, the simulation environment is described as well as results for the example occluding 3D object and target.

Simulation Environment

The algorithm was implemented in AML/X, an object-oriented programming language intended for use in design and manufacturing applications. The programs are run in the TGMS (Tiered Geometric Modeling System) environment (Ref. 5). TGMS provides an object-oriented programming interface to our in-house solid modeling system, GDP (Geometric Design Processor) (Ref. 6), as well as many geometry classes and methods.

In this framework, the occluding and target objects as well as the viewing and occluded regions, are represented as solids and any operations on them (e.g. convex hull, boolean set operations), are conveniently developed.

Simulation Results

In Figure 9 the occluding polyhedron and the target are shown. The occluding polyhedron is first decomposed into faces. Each face that lies "above" the target (i.e. in the half-space

defined by the target and the outward pointing normal to the target) is then treated as a separate occluding polygon.

Consider face F_{top} in Figure 9. This occluding face is first decomposed into its two hole loops LH, SH, and the outer material loop M (see Figure 19). Following this loop decomposition, the hole loop LH is decomposed into convex parts, namely the hole polygon LH_{ch} (the convex hull of LH), and the material polygon LH_g (a gulf of LH), while the small hole loop SH and the material loop M are already convex.

These decompositions generate four convex viewing subproblems, namely the computation of:

1. The viewing region through the small hole, V_{SH}.
2. The viewing region through loop LH_{ch}, V_{LHch}.
3. The region occluded by the material loop LH_g, O_{LHg}.
4. The region occluded by the material loop M, O_M.

The final occluded region for face F_{top} is given by:

$$O_f = O_M - (V_{SH} + (V_{LHch} - O_{LHg}))$$

and is shown in Figure 10 .

The union of this partial result with the occluded regions of the remaining faces of the polyhedron that lie above the target produces the final occluded volume of the polyhedron. A 3-D view of the final occluded region is shown in Figure 11. The visibility regions correspond to viewing the target through the small hole SH and the large hole LH. It should be observed that the regions corresponding to viewing the target through areas LH_1 and LH_2 of the large hole (Figure 19), quickly diminish as the distance from the target increases.

Figure 12 shows the perspective view of the occluding object and target for a viewpoint inside the computed viewing region (i.e. point B of Figure 11). It is seen that the target is clearly visible. Point B will also be chosen as one viewpoint for the actual camera placement experiments that are discussed in the following section.

Real Experiments

This section describes the setup used in the camera placement experiments with the hand-eye system, along with the method used to position the manipulator. Finally, the results are presented showing actual camera images of the target from selected vantage points.

Setup Description

The experimental setup is shown in Figure 14. A Javelin CCD 480 x 384 camera is fastened to the last joint of an IBM Clean Room Robot (CRR). The CRR has two manipulators, each with seven joints, three linear joints (x,y,z), three rotary joints (roll, pitch and yaw) and the gripper joint.

The three-dimensional occluding object and the target are shown in Figure 13. This object is assembled from smaller primitive objects (i.e. cubes, parallelpipeds etc.) so that it can be reconfigured to test a variety of occlusion arrangements.

Experimental Procedure

A three-dimensional solid model of the object is built in the TGMS/GDP environment and the occluded region associated with the chosen target is generated by the visibility algorithm. The generation of the occluded region of the top face of the object alone is shown in

Figure 10, while the occluded region for the whole object is shown in Figure 11. Both were explained previously in "Simulation Results". After the occluded region is computed, viewing positions are chosen, inside and on the boundary of the visibility regions.

The manipulator with the mounted camera is used to place the camera at the chosen viewing positions with respect to the occluding object and target. Each camera position chosen, is known only with respect to an object coordinate system. What needs to be determined is the manipulator location that places the camera at the chosen position. This manipulator location can be computed from the hand-eye relationship and the pose of the object in the robot world coordinate system. The hand-eye relationship is determined by using the calibration scheme of Tsai and Lenz (Ref. 13), while the object pose in the robot world is found by first computing the position and orientation of the object with respect to the camera (Ref. 9,10,11,12) and then converting this into a location expressed in the robot world coordinate system. Details regarding the camera placement computations can be found in the Appendix.

Experimental Results

The visibility regions generated for the object and target of Figure 13 are shown in Figure 11. Two views are considered for each visibility region. One view is the *comfortable view* where the target is viewed with some margin of clearance between the occluding object and the target. Another view is the *critical view* where the occluding object just barely clears the target. The purpose for choosing the critical view is for validating the preciseness of the boundary of the viewing region.

The camera locations chosen are shown in Figure 11. A and B are comfortable viewpoints, viewing the target through the small and large hole respectively. C and D are the corresponding critical viewpoints for A and B. Figure 14 shows the manipulator placed at viewpoint A and oriented towards the target center, while Figure 15 depicts the associated scene from the camera. The camera views of the target from points B, C and D are shown respectively in Figures 16, 17 and 18. Comparing Figures 15 and 17, it can be seen that the target is visible in both cases, but Figure 17 shows the edge of the small hole occluding a target edge along the boundary of the visibility region. This edge-edge interaction however cannot be seen in the critical view from D (see Figure 18) because the camera placement error is magnified by the oblique viewing angle. In Figure 16, the target is visible but out of focus, indicating that other constraints (e.g. depth of field) need be considered in the general task of sensor placement planning.

Conclusion

Camera placement results for a robotic hand-eye system are presented based on a visibility planning algorithm that avoids occlusion of a chosen visual target. The algorithm generated viewing regions for an occluding object and target and then viewpoints were selected in these regions to place a camera in a hand-eye configuration. The manipulator pose was computed to achieve proper camera position and orientation and the camera images were observed to validate occlusion-free placement.

We will extend the planning algorithms to include satisfaction of other constraints, such as resolution, field of view and focus and demonstrate results by planning zoom, focus and aperture settings for an off-the-shelf zoom lens so that these constraints are satisfied. The need for other such planning functions that satisfy task constraints other than occlusion will motivate future work and will result in more intelligent and autonomous machine vision applications for the future.

References

1. Cowan C., and Kovesi P., 1987, Automatic Sensor Placement from Vision Task Requirements, *SRI report*, Menlo Park, CA, June 1987.
2. Sakane S., Sato T., and Kakikura M., 1987, Model-Based Planning of Visual Sensors Using a Hand-Eye Action Simulator System: Heaven, *Electrotechnical Laboratory report*, MITI, Japan, 1987.
3. Tsai, R. Y., and Tarabanis, K., 1989, Occlusion-Free Sensor Placement Planning, *Proceedings of Third Annual Machine Vision Workshop*, New Brunswick, NJ, April 3-4, 1989.
4. Tarabanis K., and Tsai R.Y., 1989, Viewpoint Planning: the Visibility Constraint, *Proceedings of DARPA Image Understanding Workshop*, Palo Alto, CA, May 22, 1989.
5. Dietrich W., Nackman L.R., Sundaresan C.J., Gracer F., 1988, TGMS: An Object-Oriented System for Programming Geometry, *IBM Research Report*, IBM T.J. Watson Research Center, Yorktown Heights, NY, January 1988.
6. Wesley M. A., Lozano-Perez T., Lieberman L. I., Lavin M. A., Grossman D. D., A Geometric Modeling System for Automated Mechanical Assembly, *IBM Journal of Research and Development*, January 1980.
7. Preparata, F., and Shamos M., *Computational Geometry*, Springer Verlag, 1985.
8. Lenz, R., and Tsai, R. Y., 1988, Calibrating a Cartesian Robot with Eye-on-Hand Configuration Independent of Eye-to-Hand Relationship, *Proceedings of IEEE Computer Vision and Pattern Recognition*, Ann Arbor, MI, June 5-9.
9. Lenz, R. and Tsai, R., 1987, Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology, *Proceedings of IEEE International Conference on Robotic and Automation*, Raleigh, NC. Also to appear in IEEE Trans. on PAMI.
10. Tsai, R., 1987, A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology using Off-the-Shelf TV Cameras and Lenses, IEEE Journal of Robotics and Automation, Vol. RA-3, No. 4, August. A preliminary version appeared in 1986 IEEE International Conference on Computer Vision and Pattern Recognition, Miami, Florida, June 22-26.
11. Tsai, R., and Lenz, R., 1987A, Review of the Two-Stage Camera Calibration Technique plus some New Implementation Tips and New Techniques for Center and Scale Calibration, *Second Topical Meeting on Machine Vision*, Optical Society of America, Lake Tahoe, March 18-20.
12. Tsai, R., and Lenz, R., 1988A, Review of RAC-based Camera Calibration Vision, Society of Manufacturing Engineers, Nov.
13. Tsai, R. Y., and Lenz, R., 1988B, Real Time Versatile Robotics Hand/Eye Calibration using 3D Machine Vision, *International Conference on Robotics and Automation*, Philadelphia, PA, April 24-29.
14. Fu K.S., Gonzalez R.C., Lee C.S.G., 1987, *Robotics: Control, Sensing, Vision, and Intelligence*, New York, McGraw-Hill.
15. Hansen C., and Henderson T., 1988, Toward the Automatic Generation of Recognition Strategies, *International Conference on Robotics and Automation*, Philadelphia, PA, April 24-29.

Appendix

The camera placement computations are presented here in more detail.

The cartesian coordinate systems and homogeneous transformation matrices that are used are:

- G, the gripper coordinate system which is fixed on the robot gripper
- C, the camera coordinate system. That is, the coordinate frame fixed on the camera and centered at the lens center, with the z axis coinciding with the optical axis, and the x, y axes parallel to the image X, Y axes.
- O, The object world coordinate frame. This is an arbitrarily selected coordinate frame relative to which the coordinates of each point on the object are known.
- W, the robot world coordinate frame. It is fixed in the robot workstation, and as the robot arm moves, the encoder output of all the robot joints determines where the gripper is relative to W.
- H_{gw} defines the coordinate transformation from G to W.
- H_{oc} defines the coordinate transformation from O to C.
- H_{cg} defines the coordinate transformation from C to G.

In this framework, the manipulator location needed is expressed by H_{gw} , since, given the position and orientation of the gripper as H_{gw} and the robot calibration, the corresponding manipulator joint values can be found to position the gripper as desired. H_{gw} can be determined knowing the corresponding homogeneous transformations H_{wo} , H_{oc} and H_{cg}, using:

$$H_{gw} = H_{wo}^{-1} H_{oc}^{-1} H_{cg}^{-1} \tag{3}$$

In other words, the gripper to robot world relationship can be found when the pose of the object in the robot world (H_{gw}), the pose of the camera with respect to the object (H_{oc}) and finally the camera to gripper relationship (H_{cg}) are all known.

Each homogeneous transformation matrix in the right hand side of (3) can be computed as follows:

Computation of H_{cg}

H_{cg} is obtained by performing hand-eye calibration using the method of Tsai and Lenz (see Ref. 13).

Computation of H_{oc}

H_{oc} is determined by the location and orientation chosen to place the camera with the respect to the object coordinate system, O. The location is given by the point chosen to place the camera, while the orientation is taken to be such that the optical axis intersects the target center so that the target may be within the field of view of the camera. The extra degree of freedom for the orientation, corresponding to the rotation of the xy image plane of the camera around the optical axis, is utilized to obtain feasible manipulator positions. Given the location and orientation of the camera coordinate system with respect to the the object coordinate system, the homogeneous matrix H_{oc} can be computed from:

$$H_{oc} \equiv \begin{bmatrix} R_{oc} & T_{oc} \\ 0\ 0\ 0 & 1 \end{bmatrix}$$

where the rotation matrix R_{oc} and the translation vector T_{oc}, are given by:

$$R_{oc} = \begin{bmatrix} i_c \cdot i_o & i_c \cdot j_o & i_c \cdot k_o \\ j_c \cdot i_o & j_c \cdot j_o & j_c \cdot k_o \\ k_c \cdot i_o & k_c \cdot j_o & k_c \cdot k_o \end{bmatrix} \quad , \quad T_{oc} = \begin{bmatrix} T_{oc_x} \\ T_{oc_y} \\ T_{oc_z} \end{bmatrix}$$

where (i_c,j_c,k_c), and (i_o,j_o,k_o) are the unit vectors along the axes of the camera C and object O coordinate sytems, and T_{oc} is the position vector of the chosen viewpoint in the object coordinate system (Ref. 14).

Computation of H_{ro}

H_{ro} represents the pose (position and orientation) of the object in the fixed robot world coordinate system. This relationship can be determined from an arbitrary view of the object as it lies in the workcell, when the gripper and camera poses as well as the camera-gripper relationship are all known. For this view, H_{ro} can be computed by:

$$H_{ro} = H_{oc}^{-1} H_{cg}^{-1} H_{gw}^{-1} \tag{4}$$

a relationship equivalent to (3). H_{cg} is again obtained by performing hand-eye calibration. H_{gw} is given by the robot as the location of the gripper within the robot workcell (Ref. 8). H_{oc} is obtained by computing the extrinsic calibration parameters of the camera with respect to the object coordinate system. The camera calibration method by Tsai was applied (Ref. 9,10,11,12) using features of the object rather than a calibration pattern. The object features chosen for the part shown in Figure 13, were the centers of the circles on the top face of the object. The association between image and object features that is required for the extrinsic calibration of the camera is considered known.

At this point, having computed H_{ro}, the manipulator position, H_{gw}, can be found from (3), and the robot can be commanded to the corresponding location and orientation.

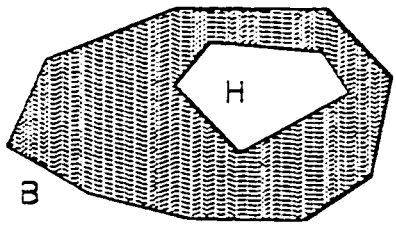

Figure 1. The occluding object is a polygon with a hole in it.

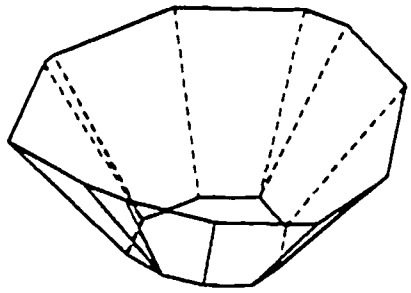

Figure 2. The computed occluded region of B.

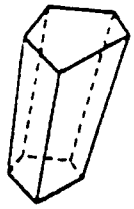

Figure 3. Viewing region for H.

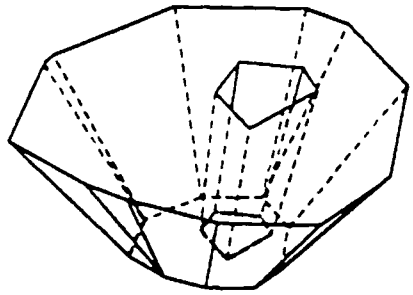

Figure 4. The resultant occluded region.

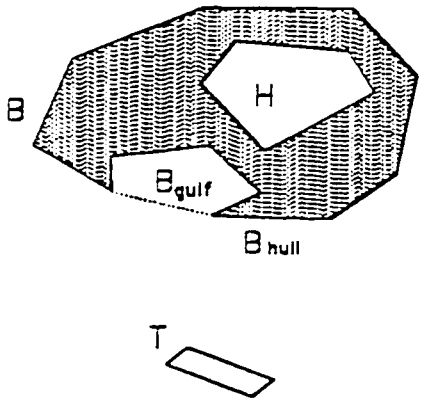

Figure 5. The occluding object is a concave polygon with a hole in it.

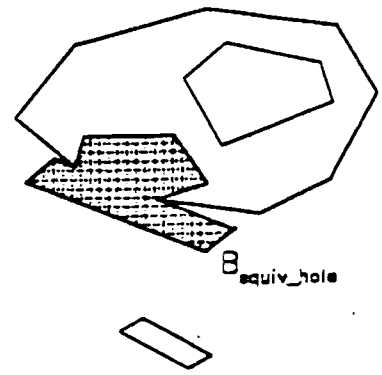

Figure 6. The gulf is extended to become a virtual gulf or equivalent hole.

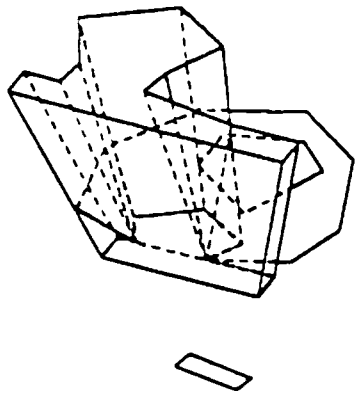

Figure 7. The viewing region through the equivalent hole.

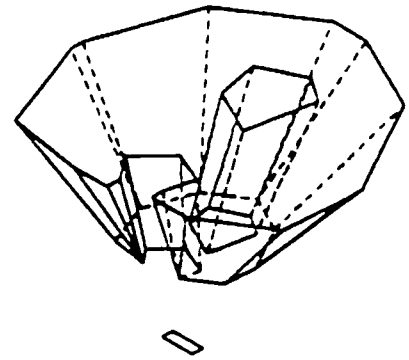

Figure 8. The resultant occluded region.

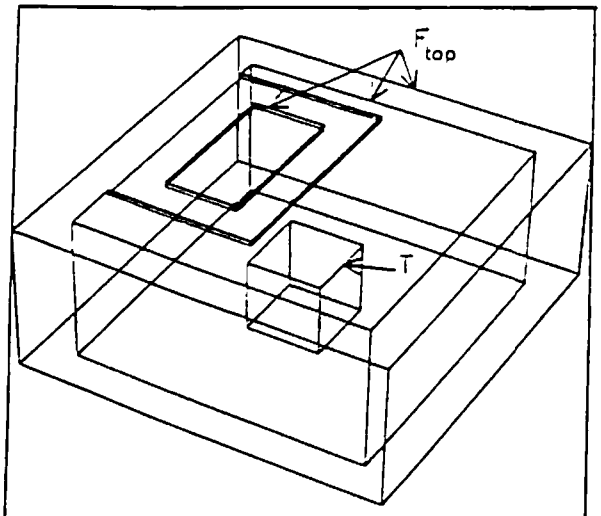

Figure 9. CAD model of the object and target.

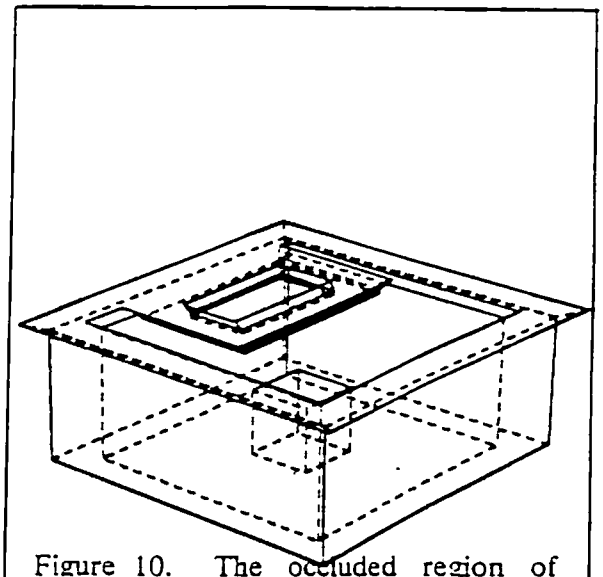

Figure 10. The occluded region of F_{top}.

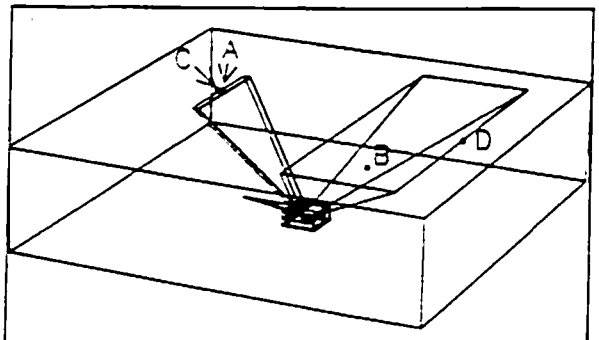

Figure 11. The final occluded region of the object.

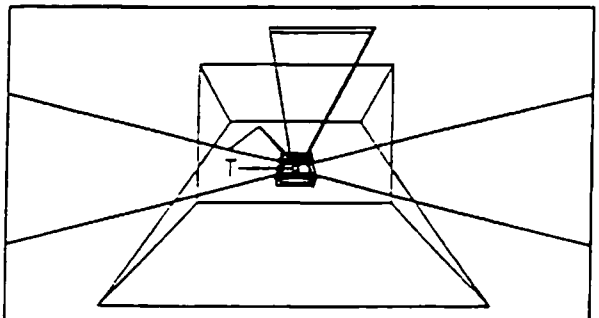

Figure 12. The view of the target from viewpoint B.

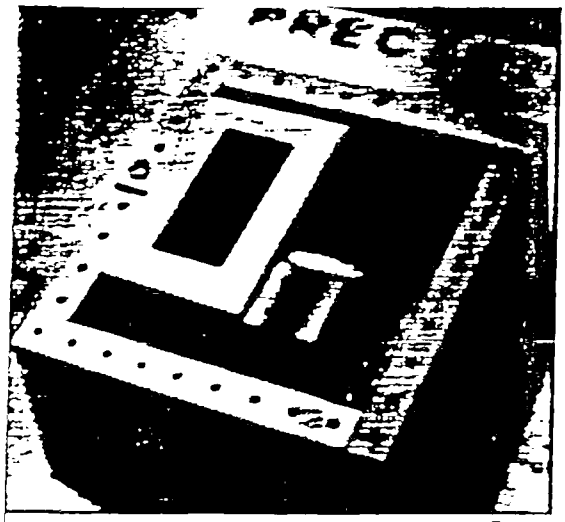

Figure 13. The actual object and target.

Figure 14. The experimental setup.

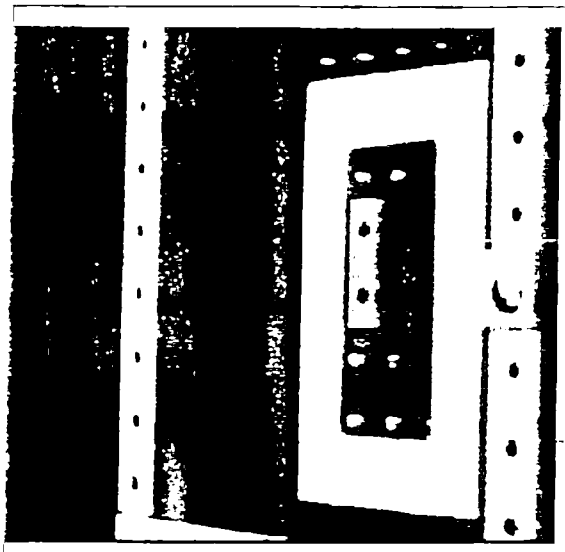

Figure 15. The comfortable view of the target from viewpoint A.

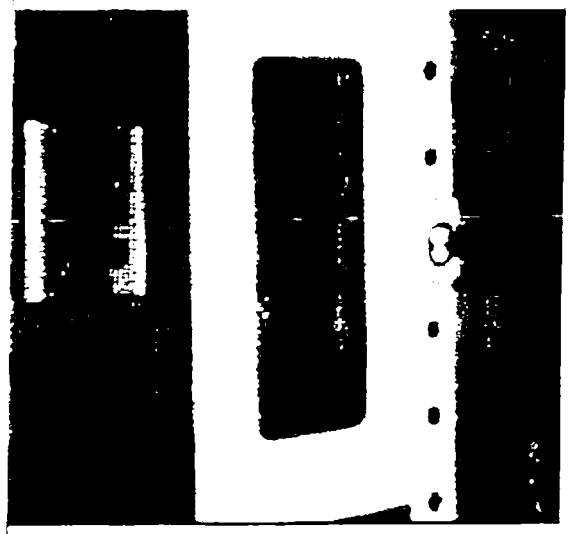

Figure 16. The comfortable view of the target from viewpoint B.

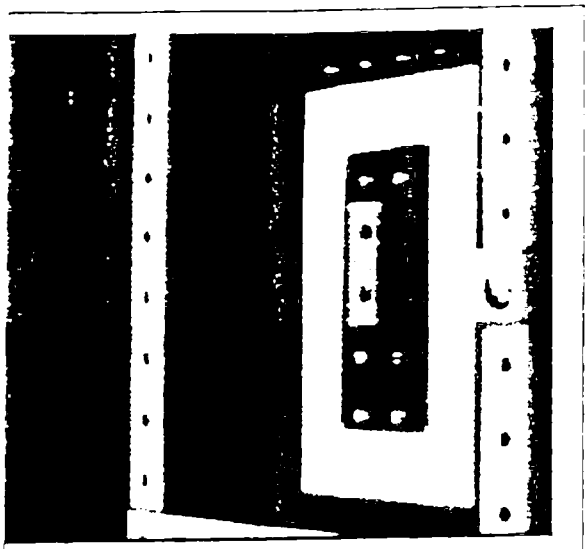

Figure 17. The critical view of the target from viewpoint C.

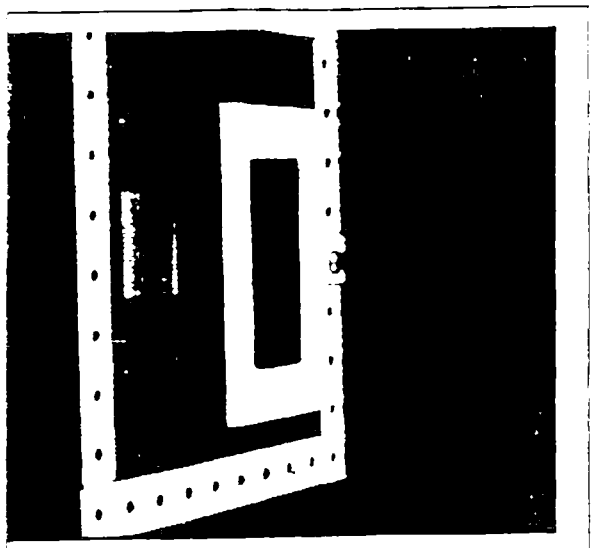

Figure 18. The critical view of the target from viewpoint D.

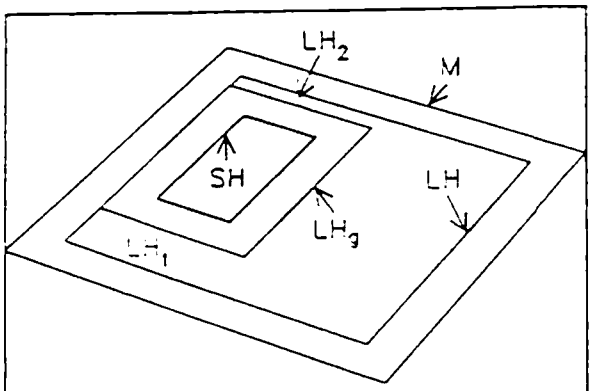

Figure 19. The decompositions of F_{top}.