To appear in
Dynamics of Continuous, Discrete and Impulsive Systems
http:monotone.uwaterloo.ca/∼journal

A Graphics Hardware-based Accessibility Analysis for Real-time Robotic Manipulation

Han-Young Jang[1], Hadi Moradi[2], Sukhan Lee[2], Daesik Jang[3], Eunyoung Kim[2] and JungHyun Han[1]

[1]Department of Computer Science and Engineering
Korea University, Seoul, Korea

[2] School of Information and Communications Engineering
Sungkyunkwan University, Suwon, Korea

[3] Department of Computer Information Science
Kunsan National University, Kunsan, Korea

Abstract. This paper presents a new approach to real-time accessibility analysis for robotic manipulation. The workspace is captured using a stereo camera, and processed into a 3D model which is composed of extracted planar features, recognized objects, and unrecognized 3D point clouds organized using an octree. When a robot is requested to manipulate a recognized object, the local accessibility information for the object is retrieved from the object database. Then, the accessibility analysis procedure is invoked to verify the local accessibility and determine the global accessibility. The verification process utilizes the visibility query supported by graphics hardware. The experimental results show the feasibility of real-time accessibility analysis using commodity graphics hardware and its performance gain.

Keywords. accessibility analysis, visibility, graphics hardware, robotic manipulation, motion planning

1 Introduction

Mobile robots have received a lot of attention in the areas of service and personal assistance, especially for aiding the elderly or disabled. The research work presented in this paper aims at a mobile home service robot, which is requested to access, grasp and remove predefined objects. For this purpose, we have designed and implemented a motion planner, composed of a 3D workspace modeling module, an accessibility analysis module, and a path planning module. This paper focuses on the geometric reasoning algorithm of the accessibility analysis module, which has been built upon the 3D workspace modeling techniques of [6] and integrated with a potential field path planner [2]. The role of the accessibility analysis module is to determine the directions along which the robot gripper can access and remove the requested objects.

The success of the mobile home service robots depends on their real-time performance. Unfortunately, software-based approaches to accessibility analysis would be slow, and therefore often are inappropriate for real-time manipulation. The accessibility analysis algorithm proposed in this paper utilizes a key function of commodity graphics hardware, the visibility query, and guarantees real-time performance.

The organization of this paper is as follows. Section 2 surveys the related work. Sections 3 and 4 present the workspace modeling and the accessibility representation, respectively. Section 5 discusses the spatial reasoning algorithms for accessibility analysis. Section 6 presents the experimental results, and evaluates the performance of the proposed approach. Finally, Section 7 concludes the paper.

2 Related Work

Accessibility analysis refers to a spatial reasoning activity that seeks to determine the directions along which a tool can access an object. The traditional application fields include automatic inspection with coordinate measuring machines (CMMs) [1][9], tool path planning for assembly [12], sensor placement for computer vision [11], numerically controlled (NC) machining [4], etc.

The majority of the work in accessibility analysis has been done in the inspection field. Spyridi and Requicha [10] were the first to incorporate a systematic accessibility analysis for the features to be inspected. They use a computationally intensive method to determine if a point is locally accessible and then verify it considering the entire workpiece.

An accessibility analysis approach, for an infinite length probe, based on a ray tracing algorithm was proposed by Lim and Menq [7]. They determine a discrete 3D accessibility cone which is transformed into a 2D map where only the orientation of the probe is expressed by two angles in a spherical coordinate system. A heuristic is used to determine the optimal probe direction for a set of points to be inspected .

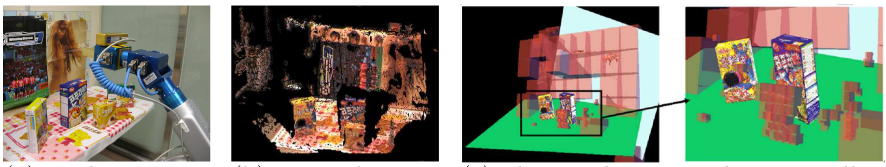
Limeiam and ElMaraghy [8] addressed accessibility analysis of a point in 3D space using elementary solid modeling operations such as intersection, translation and scaling. The method determines an accessible point, and an extended version of the method can be used for surface accessibility.

In the studies centered on inspection, it has been generally assumed that the environment is open for a probe's motion and only the workpiece may collide with the probe. Moreover, virtually all methods have proposed algorithms that run mostly off-line to determine the accessibility. Such methods are not appropriate for real-time manipulation in a cluttered environment.

In computer graphics, visibility has been a fundamental problem since the very beginning of the field. Among the visibility issues, the focus was dominantly on hidden surface removal. The problem has been mostly solved, and the z-buffer [3] technique dominates for interactive applications. In addition to the z-buffer, current commodity graphics hardware supports an image-space visibility query that checks whether a primitive is visible or not. This paper reports an accessibility analysis algorithm based on the visibility query and its application to robotic manipulation.

3 Workspace Modeling

Environment modeling is crucial for autonomous mobile robots, especially for service robots that perform versatile tasks in everyday human life. However, real-time workspace modeling in a cluttered environment is a difficult problem, and few research results have been reported.



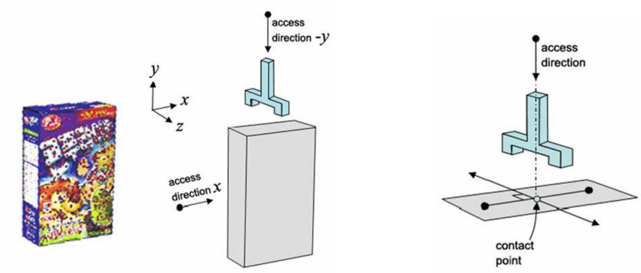
(a) workspace (b) range data (c) planes, objects, and octree cells

Figure 1: Workspace modeling

The service robot in the current study is equipped with a stereo camera mounted on a parallel jaw gripper (Fig. 1-(a)), and the stereo camera captures the range data in the form of 3D point clouds (Fig. 1-(b)). The authors of this paper proposed a new approach to real-time 3D workspace modeling [6] which extracts the global planar features and then recognizes the objects, henceforth called the *target objects*, to be manipulated. The point clouds which are not included in the planar features and the target objects are considered *obstacles* and represented in an octree. Fig. 1-(c) shows the extracted planar features, the recognized objects (two cereal boxes) and the obstacles (illustrated as octree cells).

4 Local Accessibility

In the proposed approach, all target objects have complete solid models in the database, and the database contains *local accessibility* information of each object. The accessibility information is named local in the sense that it is defined without considering the entire workspace. Fig. 2 shows the local accessibility representation for a cereal box as an example. The accessibility information specifies the *access directions* along which the gripper can access



(a) access directions (b) contact point

Figure 2: Local accessibility

the target object. With a robot arm of a small gripper, it is reasonable to define four access directions: $\pm x$ and $\pm y$ with respect to the local coordinates of the cereal box. (In Fig. 2-(a), only x and $-y$ are illustrated.) In contrast, z is not a valid access direction.

Given an access direction vector, there can be (infinitely) many *graspable* or *contact points* for an object. As illustrated in Fig. 2-(b), a contact point is defined as the intersection between the object surface and the gripper axis when the gripper approaches the object along the access direction. In the current implementation, a set of contact points is represented as a Bézier curve which is called a *contact curve*. Local accessibility is then represented as a set of <access direction, contact curve> pairs.

In the case of the cereal box shown in Fig. 2, four access directions, $\pm x$ and $\pm y$, are stored in the database. Based on the intuitive human graspability preferences, *priorities* are given to the access directions. For instance, both $+x$ and $-x$ are given the first priority and both $+y$ and $-y$ are given the second priority. Such priorities tell the robot to try either $+x$ or $-x$ first and then try either $+y$ or $-y$ when the first try fails.

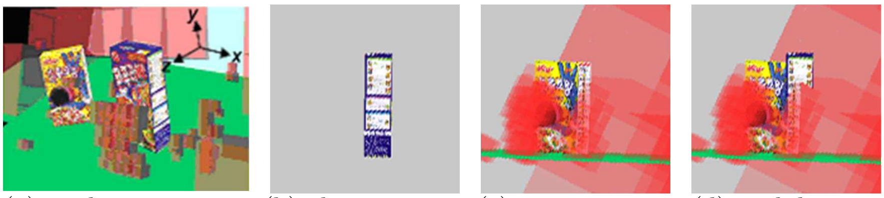
5 Global Accessibility

To be able to access and grasp an object in a given workspace, its local access directions should be verified considering the entire workspace. If a local access direction is verified, i.e. if the gripper can access the object along the local access direction, it is called *global access direction*. Geometric reasoning is required for the verification process. This section shows how the global accessibility is verified through *visibility*, which is classified into object visibility (Section 5.1) and gripper visibility (Section 5.2).

Recall that local accessibility is encoded as a set of <access direction, contact curve> pairs, and priorities are given to the access directions. The algorithm starts with the access direction with the highest priority and then tests it for global accessibility. If the test succeeds, a contact point is computed and returned, which is determined to be optimal for the current obstacle configuration. Then, the robot arm linearly translates toward the contact point along the access direction, grasps the object, and removes it along the opposite direction. If the test fails, the access direction of the next-priority is selected, and the same process is repeated.

5.1 Object Visibility

In order for an object to be accessed along a direction, the object should be *fully visible* along the direction. The visibility test is done using *visibility query* supported by commodity graphics hardware. The visibility query renders a given object and returns the number of visible pixels of the object.



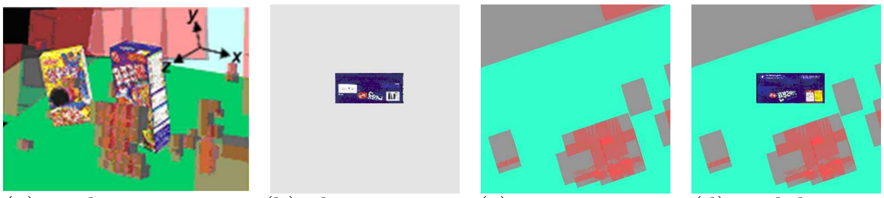
(a) workspace (b) object (c) environment (d) visibility

Figure 3: Object visibility test for $-x$

In general, two types of projection are supported by graphics hardware: *orthographic* and *perspective*. We use orthographic projection, and its viewing direction is set to the access direction. Assume the priority of $\pm x$ over $\pm y$ for the cereal box in Fig. 2. The workspace is shown in Fig. 3-(a). Let us discuss the visibility test for the access direction $-x$. First, the visibility query is issued with the target object only. Obviously, the target object is fully visible, as shown in Fig. 3-(b)[1]. The visibility query returns n, the number of pixels occupied by the target object. Second, the depth-buffer is cleared, and the environment is rendered excluding the target object, as shown in Fig. 3-(c). Finally, the visibility query is issued by rendering the target object into the environment. Then, the visibility query returns m, the number of visible pixels occupied by the target object. As shown in Fig. 3-(d), the cereal box is partially invisible due to some obstacles represented in octree cells. It is found by comparing n and m. As $n > m$ along the access direction $-x$, the object is determined to be partially invisible, and therefore *not* accessible along $-x$.

In Fig. 3, we have shown that the cereal box is not accessible along $-x$. The same object visibility test along the access direction x shows that the box is not accessible either. Then, the access directions of the next-priority, i.e. $\pm y$, are investigated. Due to the presence of the planar feature, the

[1]The rendered images are provided just for easy understanding, and are not really used for geometric reasoning. Only the *visibility query* is issued.

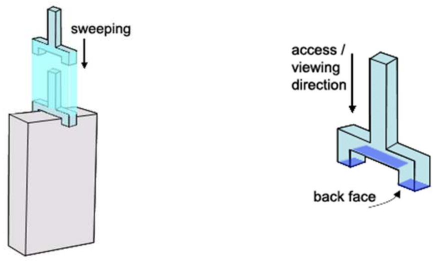


(a) workspace (b) object (c) environment (d) visibility

Figure 4: Object visibility test for $-y$

access direction y will be immediately rejected. Finally, the object visibility test will prove that the cereal box is accessible along $-y$. Fig. 4 illustrates the process of object visibility test along the access direction $-y$.

5.2 Gripper Visibility



(a) sweeping gripper (b) gripper back faces

Figure 5: Gripper visibility test with back faces

Object visibility is the necessary condition for object manipulation. The sufficient condition is that the gripper should be able to access, grasp and remove the object. If the gripper can translate towards the target object *without colliding* with any obstacle in the scene to obtain the configuration of Fig. 5-(a), where the gripper contacts the target object, the object is determined to be *globally accessible*.

The *configuration* of an object is a specification of the position of every point in the object relative to a fixed reference frame. The *configuration space* or *c-space* of an object corresponds to the space of all possible configurations of the object [5]. In accessibility analysis, we want to determine the *collision-free configuration space* of a gripper, i.e. the set of all possible configurations in which the gripper does not collide with any obstacle in the environment:

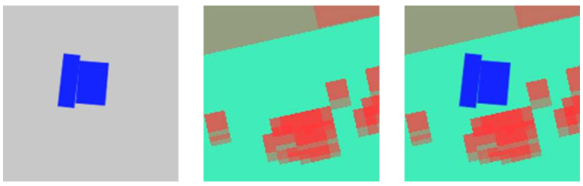
$$\{q | q \in C, G(q) \bigcap X = 0\} \tag{1}$$

where q is a configuration of the gripper, C is the c-space, $G(q)$ is the region of the workspace occupied by the gripper in configuration q, and X is the region of the workspace occupied by obstacles. Specifically in this study, the set of globally accessible directions is defined as a set of configurations in which the gripper can linearly translate toward contact points without colliding with any obstacle:

$$\{q|q \in C, p \in P, G_p(q) \bigcap X = 0\} \quad (2)$$

where q is a configuration of the gripper, C is the c-space, P is the set of all contact points of the target object, and $G_p(q)$ is the set of all configurations along the surface normal vector at contact point p. Note that $G_p(q)$ is equivalent to the *swept volume* of the gripper along the surface normal vector at p.

In principle, the global accessibility test requires the swept volume to be tested for collision with the obstacles. Sweeping and collision detection are not cheap operations. Fortunately, they can be replaced by the *gripper visibility test*. Given a viewing direction vector d (access direction of the local accessibility instance) for the orthographic projection, a boundary face f of an object is classified into a *back face* if $f_n \bullet d > 0$, where f_n is f's surface normal vector. It is sufficient to consider only the back faces of a gripper rather than the gripper's entire geometry. The gripper back faces are shaded in Fig. 5-(b). If all the back faces at the grasping pose are visible along the access direction, it is concluded that the sweeping gripper does not collide with any obstacle.



(a) gripper (b) environment (c) visibility

Figure 6: Gripper visibility test for $-y$

The gripper visibility test goes through the process similar to that of object visibility test. First, the visibility query is issued only with the gripper at the grasping pose. The gripper is fully visible as illustrated in Fig. 6-(a). The number of visible pixels n is returned by the visibility query. Second, the depth-buffer is cleared, and the environment is rendered, as shown in Fig. 6-(b)[2]. Finally, the visibility query is issued by rendering the gripper into the environment. Then, the visibility query returns m, the number of

[2]The target object does not need to be rendered because the gripper is placed on top of the object.

pixels occupied by the gripper. If $n = m$, the gripper is fully visible, the sweeping gripper does not collide with any obstacle, and the target object is determined to be globally accessible. In Fig. 6-(c), the object is determined to be globally accessible.

In general, there are (almost always) infinitely many contact points. Note that the gripper visibility can be verified for some contact points while it may not be for others. For the purpose, the contact curve is regularly sampled. For each sampled point, the gripper visibility test is done, and the optimal one among them is selected using a heuristic. Recall that the object is accessed along a globally accessible direction and then removed along the opposite direction. Fig. 6-(d) shows a snapshot of object manipulation.

6 Experiments

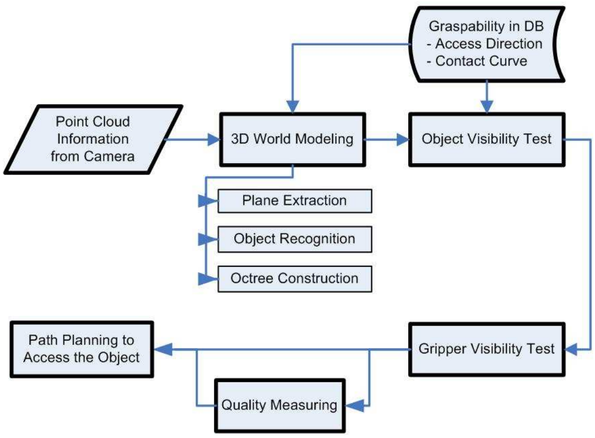


Figure 7: Object manipulation flowchart

(a) scene capture (b) initial pose (c) grasping (d) box removal

Figure 8: Demonstration

The motion planning algorithm presented in this paper was implemented on a modest PC with 2.8 GHz Pentium 4 and NVIDIA Geforce 6600GT graphics card. Fig. 7 illustrates the flowchart of the algorithm, and Fig. 8

shows snapshots of a sample manipulation task in which the scene is captured (Fig. 8-(a)), the gripper moves to the initial pose to start translation toward the object (Fig. 8-(b)), the object is grasped (Fig. 8-(c)), and finally the object is removed (Fig. 8-(d)).

module	average time (ms)
workspace modeling	325
object visibility test	25
gripper visibility test	13
path planning	98

Table 1: Performance evaluation

Table 1 shows the processing time consumed by each module of the system. Note that workspace modeling and path planning are done by CPU but accessibility analysis (the object and gripper visibility tests) is performed using graphics hardware. CPU implementation of the accessibility analysis algorithm would be slow and therefore inappropriate for real-time manipulation. The real-time performance in this study is obtained by using graphics hardware.

7 Conclusion

This paper presents a novel approach to accessibility analysis for manipulative robotic tasks: visibility-based geometric reasoning. The accessibility analysis process utilizes the visibility query, which is accelerated by graphics hardware. The performance and robustness of the proposed approach are evaluated in cluttered indoor environments. The experimental results demonstrated that the proposed methods are fast and robust enough to manipulate 3D objects for real-time robotic applications. The determined accessibility direction is used by the path planning system to generate a collision free path to grasp and lift the object.

Currently, the objects to be manipulated are limited to box-shaped ones. The proposed approach is being extended to deal with a wider range of objects such as cylindrical cans and bottles with arbitrary silhouettes. For such an extension, the proposed framework can be used without alteration. However, the authoring stage for the object database construction is complicated, so a more general representation for the contact points is being investigated.

8 Acknowledgements

This paper was performed for the Intelligent Robotics Development program, the 21st century frontier R&D programs funded by the Ministry of Commerce, Industry and Energy of Korea.

9 References

[1] ANSI, Dimensioning and Tolerancing: ANSI Y14.5M-1982, American Society of Mechanical Engineers, February 1982.

[2] O. Brock, Generating Robot Motion: The Integration of Planning and Execution, Ph.D. Thesis, Stanford University, USA, 1999.

[3] E. Catmull, A Subdivision Algorithm for Computer Display of Curved Surfaces, Ph.D. Thesis, University of Utah, USA, 1974.

[4] P. Gupta, R. Janardan, J. Majhi and T. Woo, Efficient Geometric Algorithms for Workpiece Orientation in 4- and 5-Axis NC Machining, *Computer-Aided Design*, **28**, No. 8, (1996) 577-587.

[5] J.-C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, 1991.

[6] S. Lee, D. Jang, E. Kim, S. Hong, and J. Han, A Real-time 3D Workspace Modeling with Stereo Camera, *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, August 2-6, 2005.

[7] C. P. Lim and C. H. Menq, CMM Feature Accessibility and Path Generation, *International Journal of Production Research*, **32**, (1994) 597-618.

[8] A. Limaiem and H. A. ElMaraghy, A General Method for Accessibility Analysis, *Proc. of IEEE International Conference on Robotics & Automation*, Albuquerque, USA, April 1997.

[9] S. Spitz and A. Requicha, Accessibility Analysis Using Computer Graphics Hardware, *IEEE Transactions on Visualization and Computer Graphics*, **6**, No. 3, (2000) 208-219.

[10] A. Spyridi and A. Requicha, Automatic Programming of Coordinate Measuring Machines, *Proc. of IEEE International Conference on Robotics & Automation*, San Diego, USA, May 1994, pp. 1107-1112.

[11] E. Trucco, M. Umasuthan, A. Wallace and V. Roberto, Model-Based Planning of Optimal Sensor Placements for Inspection, *IEEE Transactions on Robotics and Automation*, **13**, No. 2, (1997) 182-193.

[12] R. Wilson, Geometric Reasoning about Assembly Tools, *Artificial Intelligence*, **98**, No. 1, (1998) 237-279.

email: jhan@korea.ac.kr (JungHyun Han, corresponding author)
http://media.korea.ac.kr/