

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

Task Oriented Vision

Katsushi Ikeuchi Martial Hebert

July 1991

CMU-CS-91-163₂

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

An earlier version appeared in Proc. of 1990 DARPA Image Understanding Workshop

This research was sponsored in part by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-90-C-1465, ARPA Order No. 7597 and in part by NASA under Grant NAGW 1175.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government.

510.7808

C28r

c.2

Keywords: vision paradigm, hand-eye, grasping, shape representation, bin-picking, rock sampling, unstructured environment

Abstract

This paper overviews two recently completed vision systems (a rock sampling system for planetary rovers and a bin-picking system for industrial robots). Then, we will examine the reason why these two systems have different architectures although their goals are roughly same, picking up something by visual observation. Based on this discussion, we will develop the task-oriented vision paradigm, and examine the difference between the task-oriented vision paradigm and the traditional Marr's paradigm. We will also explore the research issues necessary for completing the task-oriented vision paradigm.

Contents

1	Introduction	
2	Rock Sampling System	
3	Bin Picking System	
4	System Analysis	
4.1	Rock-Sampling System	
4.2	Bin-picking system	
5	Task Oriented Approach	
6	Conclusion	
7	Acknowledgement	

1 Introduction

A critical issue in designing a robot vision system is how to organize the relevant components into a vision program. Such components include object representations, features, segmentation methods, image acquisition strategies, and sensors. The choice of such vision components governs the quality of resulting vision programs.

There are two approaches to the problem of organizing the components: general purpose oriented and task-oriented. Researchers in the general purpose oriented school claim that we should build the vision system to be able to solve all the vision tasks using a single architecture, that is, an architecture in which a fixed selection of components is executed in a fixed order. They also claim that we should avoid to use any task specific constraints to build a vision system.

We, researchers in the task oriented school, claim that we should prepare different architectures of vision systems and that, depending on the task (the goal of the system and the environment in which such a task is achieved), a robot vision system should change its architectures so that it has the optimal selection of the components to achieve a given task.

In the task-oriented approach, however, few attempts have been made to clarify and establish theories for the task oriented vision and methodologies to implement the theories for building a vision system of the optimal architecture beyond several ad hoc trials.

This paper proposes a task-oriented vision approach and investigates the design of vision systems in a systematic way. We will focus on the design of robotics systems that involve the localization and grasping of objects because it is a good illustration of the task oriented approach. In order to illustrate this approach, this paper will overview two vision systems recently completed: rock sampling vision for planetary rover robots and bin picking vision for industrial robots. We will illustrate the task oriented approach in designing these two vision systems. Then, we will derive a general framework for designing vision systems under the task-oriented approach.

2 Rock Sampling System

One of the most important goals of a planetary exploration mission is to collect and analyze terrain samples. As part of the CMU Ambler project [5], we are investigating techniques to collect small rocks in the sand. This section overviews the architecture of the rock sampling system.

Image Acquisition Sensor The upper left figure in Figure 1 shows a typical scene to the system. From this scene, a range image is acquired using a range finder [20]. Each pixel in a range image possesses its 3D coordinate system measured with respect to the sensor.

Segmentation From a range image, three types of features are extracted:

- shadows,
- orientation discontinuities, and
- range discontinuities.

These features give an indication of where the boundaries of the rocks may be located in the scene. These features are, unfortunately, not sufficient for reliably extracting rocks from the scene, because these features are fragmentary due to that the rocks are partially buried in the sand. Therefore, we cannot use a simple region extraction technique that would assume that the features are grouped into closed boundaries.

The existence of features are taken as a rock hypothesis. Since we know the configuration of the sensor, we can derive the approximate location of the center of a rock from the distribution of the corresponding shadow regions and depth discontinuities.

We implemented an iterative segmentation algorithm, similar to the "snake" algorithm [17]. A snake has a close deformable boundary, such as a rubber band, which is attracted by features and tries to shrink to its original size. The algorithm grows the boundary of the snake until it connects features "reasonably well".

In order to implement the attractive and shrinking forces, we use two kinds of energy field: external (attractive) and internal (shrinking).

The external energy is given by the following three forces:

- shadow attractor,
- orientation discontinuity attractor, and
- range discontinuity attractor.

Following the attractive forces, the boundary moves towards the surrounding features. At the same time, over-growing is avoided by using the internal energy of the contour. We implemented the following internal energy field:

- center attractor and
- region attractor.

A snake is attracted by its center position. It is also attracted by itself; a snake tries to form a compact shape.

A small snake is initially located at the hypothesized center of a rock region. It grows iteratively while deforming its shape until it sits along the features where the external forces from the features and its own internal forces are in equilibrium. The bottom left figure in Figure 1 shows the schematic diagram of our segmentation algorithm.

The segmentation result in Figure 1 shows the rock region, shown in blue, which has been extracted by this algorithm. Because this approach interpolates the missing gaps between features, it allows us to locate rocks in the scene even when only a very small number of visual features are extracted from the image. This departs from other vision systems which implicitly assume that strong and reliable features can always be extracted, and therefore would not perform well in the type of unstructured environment that we are considering.

Representation In order to grasp a rock, we need parameters based on 3 dimensional information of a rock, such as the mass center and the inertia axis direction. A snake segmentation provides us the 2 dimensional contour of a rock. The range data within the extracted contour give the 3 dimensional shape of a rock. However, it only gives the 3D shape of one side (visible side) of the rock. We have to infer the whole shape of a rock from this visible side shape for obtaining the 3D parameters necessary for grasping.

We will use a superquadric surface to approximate the whole rock shape from the range data. A superquadric is a generalization of an ellipsoid that can represent a wide variety of shapes with a small number of parameters [19, 3].

$$\left(\left(\frac{x}{a}\right)^{e_2} + \left(\frac{y}{b}\right)^{e_2}\right)^{e_1} + \left(\frac{z}{c}\right)^{e_1} = 1 \quad (1)$$

where a, b, c are the size parameters. By changing the parameters, e_1 and e_2 , we can represent several shapes as shown in the bottom center of Figure 1.

We chose superquadrics as our representation for two reasons:

- Superquadrics are appropriate for blob-like shapes.
- Fitting superquadrics to a set of points from a partially visible object gives an estimate of the whole shape of the object, whereas more local surface representations would provide a representation of only the visible part of the object.

We implemented a standard gradient descent method to fit a superquadric surface to range data [21]. The upper center figure in Figure 1 shows a superquadric representation of the rock.

Grasping Strategy The superquadric fitting module provides the following parameters of a rock:

- mass center position,
- size, and
- axis direction.

Once an object has been represented by a superquadric, the system examines its size parameters to decide whether the rock is small enough to be picked up by the gripper.

If so, the grasping strategy is set up so that the gripper orientation direction is aligned with the rock inertia axis direction, and the gripper approach direction is along the z axis and goes through the mass center of the rock. See the grasp strategy in Figure 1.

This configuration yields the minimum potential field given by the relationship between the gripper and the rock represented by the superquadric. The upper right figure in Figure 1 shows the actual grasping.

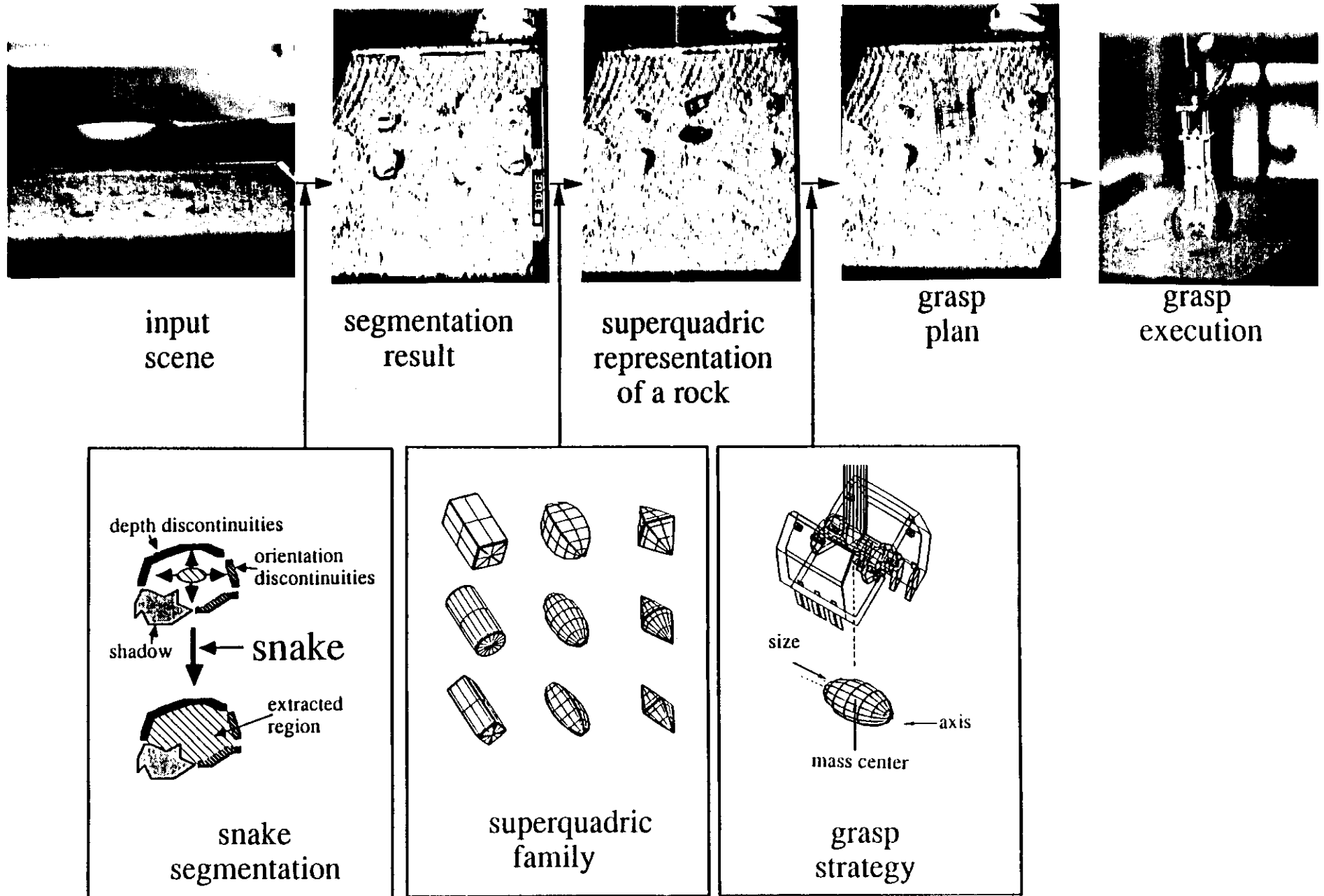


Figure 1: Rock sampling system

3 Bin Picking System

The bin-picking task is defined as picking up the top most object from a pile of the same kind of objects randomly oriented. The input scene in Figure 2 shows a typical example of a bin.

We have developed a bin-picking system using the CMU vision algorithm compiler for object localization [14, 11]. The central issue in the research is how to build the compiler, which automatically converts a geometric and sensor model into a localization program. In this section, however, we will emphasize the architecture of the run-time system generated by the compiler rather than the compiling techniques.

Image Acquisition Sensors The range data is acquired using a dual photometric stereo system [13].

Segmentation From a range image, two types of features are extracted:

- shadows:
A photometric stereo system projects three lights onto scene; each light generates a shadow region around an industrial part. Since we distribute three lights in a triangle shape, the part located at the top of the bin is surrounded by shadow regions.
- orientation discontinuities:
From a geometric model of an industrial part, we can determine the angle between two adjacent faces. We can find the minimum angle among these adjacent angles, and use this angle as the threshold for determining orientation discontinuities.

A simple segmentation method based on shadows and orientation discontinuities works quite well in this case due to the characteristics of the scene; in a bin of industrial parts, each part is enclosed by a clear occluding boundary. It is not necessary to use a more detailed segmentation method such as the snake segmentation method in the previous system. The segmentation result in Figure 2 shows the one given by the simple segmentation method; the parts (dotted regions) are separated from each other by shadows and orientation discontinuities (white regions).

The highest region is determined among the regions extracted by the segmentation program. From this highest region, the object localization process begins. This is because the highest region usually corresponds to the top-most part, and the top-most part is usually the most easiest one to pick up.

Representation The object localization process is performed by the program which is generated by the CMU vision algorithm compiler. The compiler automatically generate a localization program from the object and sensor model. The program generated can perform object localization in the least amount of computational time among several possible localization programs. See the bottom left figure in Figure 2.

Several geometric features such as area, inertia and distance between two adjacent regions, are extracted by the localization program in the predetermined order by the compiler. The program compares the extracted features with those from the model and determines the attitude and position of the part.

Using the resulting position and attitude, the program generates a part representation using the geometric model as shown in the recognition results in Figure 2. The neighboring regions are represented by dodecahedral prisms. These dodecahedral prism representations are used for collision check while constructing a grasping strategy.

Grasping strategy The grasp configuration should satisfy the following two conditions [16]:

- It should produce a mechanically stable grasp, given the gripper's shape and the part's shape. Such configurations will be called *legal grasp configuration*.
- The configuration must be achieved without collisions with other parts. Grasp configurations are limited by the relationship between the shape of the gripper and the shapes of neighboring obstacles. Such configurations will be called *collision-free grasp configuration*.

In compile mode, possible legal grasp configurations are compiled and stored at each representative attitude of the part in a grasp catalogue as shown in the bottom center figure in Figure 2.

In execution mode, the system determines to which representative attitude the current attitude of the top-most part belongs. Then, the legal grasp configurations corresponding to the representative attitude are retrieved from the grasp catalogue. These configurations are then converted into the world coordinate system based on the observed configuration of the industrial part.

The system has to find a collision-free grasp configuration among these configurations. It generates a cube representation corresponding to the work-space of each legal grasp configuration in the geometric representation as shown in the collision check in Figure 2. Then, the system examines whether the intersection exists between the gripper work space cube and the obstacle prisms in the geometric representation.

Among the possible collision free configurations found by the system, the optimal configuration (currently the one nearest to the part's mass center) is chosen. The system picks up the industrial part using the configuration as shown in upper right most figure in Figure 2.

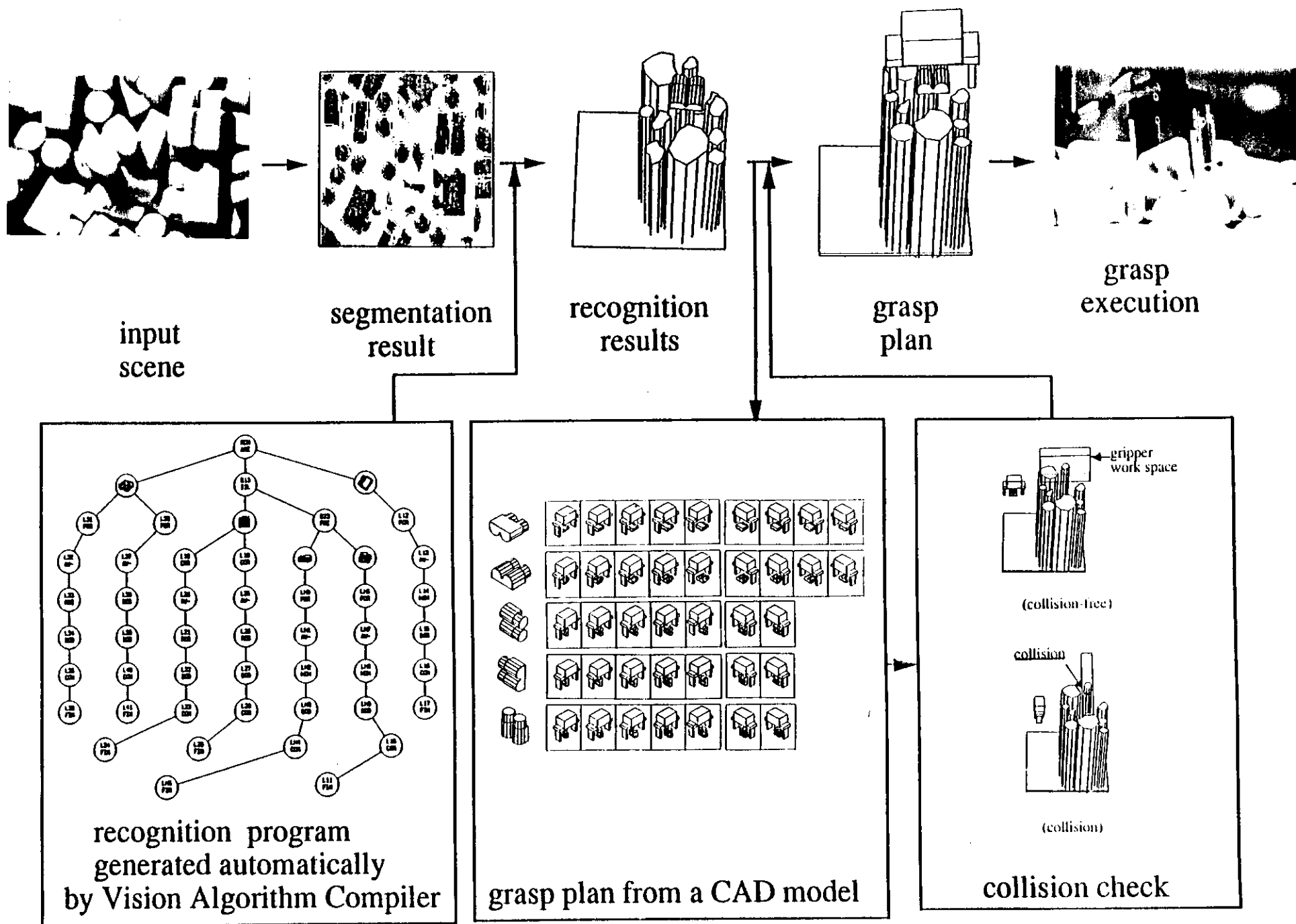


Figure 2: Bin picking system

4 System Analysis

The tasks, the two systems, rock-sampling and bin-picking, aim to achieve, are roughly same; observing a scene, determining a grasp strategy and picking up something based on the analysis. However, the two systems have completely different architectures. This section will examine the reason why such different architectures are necessary.

4.1 Rock-Sampling System

Figure 3 shows the design flow of the rock sampling (RS) system. The design begins from the task specification of the rock sampling through the image acquisition method.

Task specification The task of this system is to grasp a rock in the sand under the following conditions:

- The rocks are far enough away from each other. It is not necessary to consider the collision between the gripper and the neighboring rocks, when picking up a rock.
- We can allow the collision between the gripper and the neighboring sand. This is because
 - damaging the neighboring sand grains is not important,
 - the collision between the gripper and neighboring sand does not cause the configuration change of the rock.
- We do not know the exact shape of a rock beforehand.

Grasping Under this task specifications, it is appropriate to use a spherical grasping. See the bottom left figure in Figure 3. This grasping has the characteristics that

- it requires a large empty volume around an object to be grasped, because all the fingers approach the object from all directions.
- it may grasp the neighboring materials of the object, if any, as well as the object,

- it does not require the precise attitude and position of the object, because it grasps an object as if it wrapped the object.

In order to realize this spherical grasping, we built a clam-shell gripper. See the bottom right figure in Figure 3.

Representation Using a clam-shell gripper imposes two constraints on the representation:

- the expected mass center of a rock should be inside of the gripper
- the expected size of a rock should be smaller than the inner hull of the gripper

While working from only a partial observation of a rock and without any prior knowledge of the rock shape, we still need to recover the above information. We do not need to recover a precise shape representation of a rock, however. From this consideration, the superquadric representation was chosen, because it is described by a few parameters which can be recovered by using a fitting method such as the gradient descent method.

Segmentation For a rock partially buried in the sand, orientation discontinuities and depth discontinuities are small. Thus, it is usually difficult to detect these discontinuities reliably and extract a closed boundary based on them.

Because there is no a priori rock model available, the segmentation cannot be guided by a model as in the bin-picking system. The only available information is that a rock forms a closed boundary. Along this closed boundary, the following three boundary elements exist:

- shadow boundaries
- orientation discontinuities
- depth discontinuities

Thus, it is necessary to use a segmentation method which connects these boundary elements and extracts a closed boundary. For this purpose, a model-based segmentation method based on the snake algorithm described in Section 2 was employed.

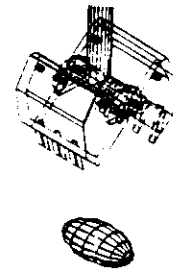
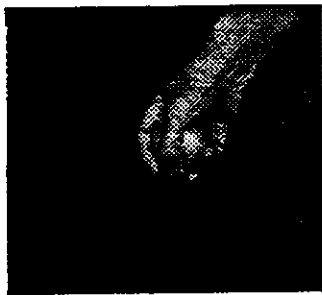
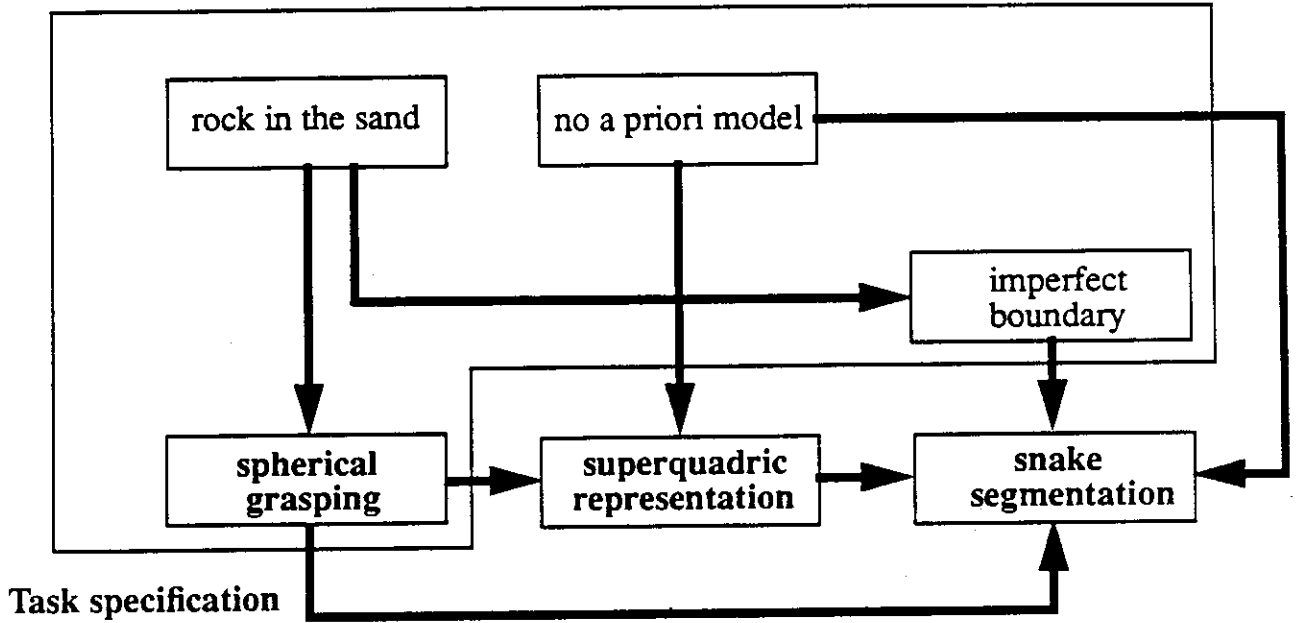


Figure 3: Design flow of Rock-sampling System

4.2 Bin-picking system

Figure 4 shows the design flow of the bin-picking (BP) system. The design flow starts with the task specification of the bin-picking and specifies all the components down to the image acquisition method.

Task-specification The task of this system is to grasp the topmost industrial part in a bin of parts under the following conditions:

- The parts are close to each other. Some collisions may occur between the gripper and the neighboring parts if a random grasping strategy is chosen.
- It should be avoided to have the collision between the gripper and the neighboring parts. This is because
 - the collision may cause damage to the neighboring parts,
 - the collision may cause configuration change of the part to be grasped, because the part is supported by the neighboring parts, and thus, it may fail to grasp the part.
- The exact shape of a part is known beforehand.

Grasping Under these task-specifications, it is appropriate to use a tip grasping. See the bottom left figure in Figure 4. This grasping has the characteristics that

- it requires only a small volume around an object to be grasped compared to other grasping strategies because only two fingers approach the object from two opposite directions
- it grasps only the object
- it requires the precise attitude and position of the object, because grasping occurs as the contact of two fingers at the same time

In order to realize this tip grasping, we built a parallel-jaw gripper. See the bottom right figure in Figure 4.

Representation In order to grasp a part using the parallel-jaw gripper,

- the precise position of two parallel planes should be known.

This constraint implies a polyhedral representation of the object. A sensor typically gives a partial observation of an object. A pair of parallel planes has two opposite surface normals. If one plane is visible from the sensor, it is likely that the other plane is self-occluded from the sensor. Even though the two planes are visible, it is necessary to have an n^2 search out of n observed planes. Thus, we decided not to find such parallel plane pairs at run time.

Instead, we decided to represent a part by a polygonal approximation given by a geometric model, to search such plane pairs in the representation at compile time, and to make the relationship between such pairs and observed part attitude. At run time, we concentrate on recovering the attitude of the part, and recovering the pairs attitude using this relationship.

Segmentation Distinct depth discontinuities can be observed around an object, because an industrial part sits on other parts, as opposed to the rock-sampling case in which a rock may be partially buried in the sand. Also from the geometric model of the object, the threshold value used to find surface discontinuities can be found from the minimum angle between adjacent faces.

The following facts are utilized for segmentation:

- An object boundary is surrounded by a shadow. Since the current implementation of the photometric stereo system employs three light sources, the top-most object in the bin is always surrounded by a shadow.
- The threshold value that defines the surface discontinuities can be defined by computing angle differences of every face pairs in the model.

Since these two classes of boundaries are distinct and connected, we do not need a method, such as the snake-based segmentation used in the rock sampling, to connect them.

As shown in this section, in order to construct a vision system, it is not enough to investigate algorithms for each vision modules, such as representation methods or segmentation methods, individually. It is also necessary to investigate the constraints and interactions among vision modules. Such constraints and interactions provide valid assumptions from which each vision module should be developed and the expected performance which each vision module should generate.

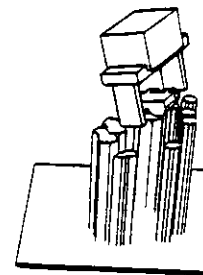
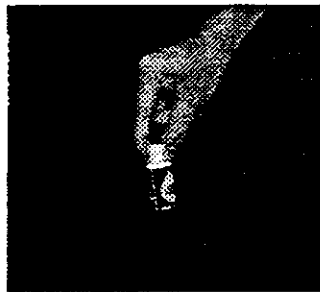
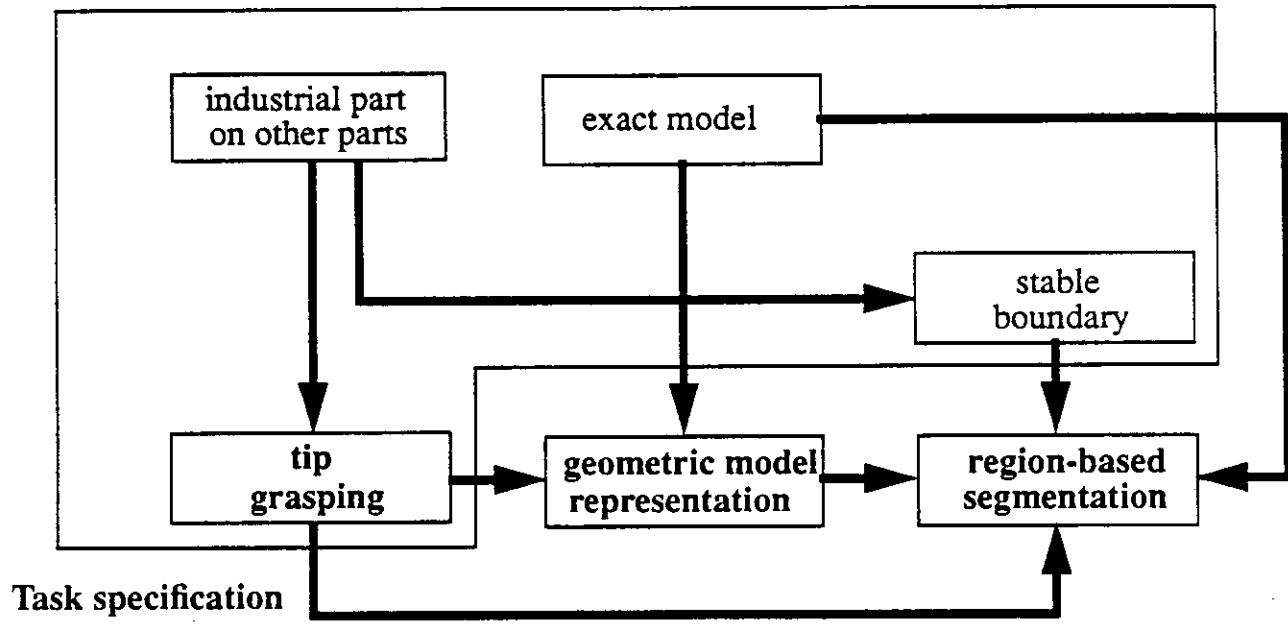


Figure 4: Design flow of Bin-picking System

5 Task Oriented Approach

An important goal for computer vision research is the development of a vision system which can serve as a complete and self-contained artificial vision unit. Currently, the majority of vision community adheres to an approach which emphasizes general purpose vision machines being constructed under the same architecture – an approach which is epitomized by Marr's paradigm. Researchers in this school try to design a general purpose vision machine which performs all vision tasks using the single architecture.

Figure 5 shows the architecture in Marr's paradigm [18]. An intermediate representation ($2 - \frac{1}{2} D$ representation) is generated from several 2D image clues such as shading, texture, and motion. Then, a final 3D representation, based on the object-centered coordinate system is generated from this $2 - \frac{1}{2} D$ representation. Independent of the nature of the tasks, the visual information is processed in a bottom-up fashion. Research focuses on each module in the system rather than on the overall system; accordingly, intermodule interactions and the system's connections to specific tasks is less emphasized.

We propose to investigate task-oriented vision systems. We assume that without aiming to see a target object (without having some specific task), we cannot see it (we cannot achieve the task). Under this assumption, we claim that one particular visual task should govern the choice of representations, vision modules, and image acquisition sensors. Thus, a task determines the optimal architecture for the vision system.

Figure 6 shows the paradigm of the system we are proposing. The basic collection of modules are the same as Marr's with the exception of a box labeled as TASK. The figure indicates that the interaction among several modules. This interaction will change as a result of the "decisions" from the TASK box and will compose the optimal architecture for the vision task at hand. We will refer to this paradigm as a task-oriented approach.

Our approach emphasizes developing not only intra-module algorithms within vision modules, which is emphasized by the traditional approach, but also inter-module interactions which depend on tasks. In other words, we consider a vision system as a whole, and under a particular task, we investigate how each vision module interacts. The key element is the logical order in which the vision components are selected and built. From the previous examples we can see that this task-oriented approach is critical in building a working system. For example, the bin-picking system would not work if a superquadric representation were used:

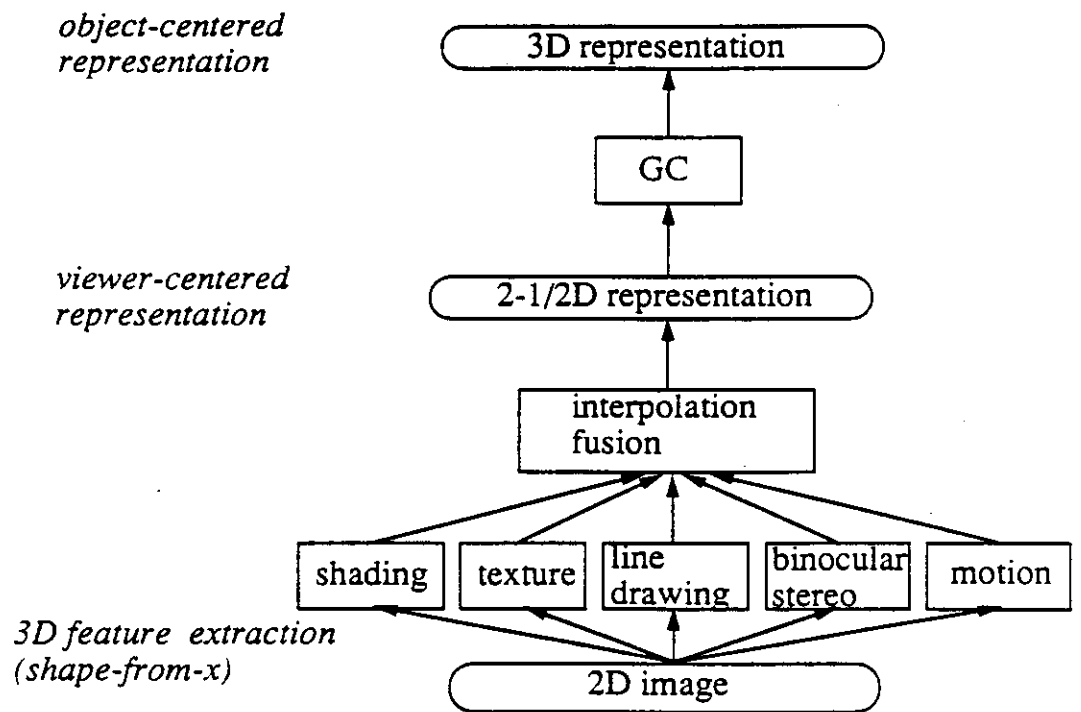


Figure 5: Marr's paradigm

The difference between the superquadric surface and the actual surface may cause the contact of one finger to occur before the other, thus causing the object to move, and even possibly to fall from the top of the bin.

In order to investigate architectures of vision systems under the task-oriented paradigm, we have to consider the following inter-module interactions:

- task specifications
- functional capabilities of a representation required by the task
- representations having such functional capabilities
- features appropriate for extracting such representations
- segmentation methods appropriate for extracting such features and representations
- image sensors and their strategies appropriate for obtaining such segmentation methods and features

Under this framework, first, we have to analyze the taxonomy of visual tasks. Figure 7 is an example of such a taxonomy. Clearly, the required representations are different for navigation and for manipulation, and the architectures of such vision systems differ accordingly. Even within the manipulation task, what to grasp and how to grasp need different architectures; what to grasp belongs to a class of object identification problems, and how to grasp belongs to a class of object representation problems.

As an example of the Task-oriented paradigm, in the remainder of this section, we will consider the issues inherent in analyzing interactions between grasping strategies and architectures. Task oriented vision starts from task-specification. In this example, task specification can be translated into grasping strategies.

Taylor and Schwarz [22] classified human grasping strategies into the following six categories:

- *Spherical grasping* – grasps an object by closing all fingers from all directions. Very stable grasping can be achieved as the contact occurs at several points on the whole surface of the object.
- *Cylindrical grasping* – grasps a cylindrical object from all directions in one plane. The contact occurs at the points along the cross-sectional circle.

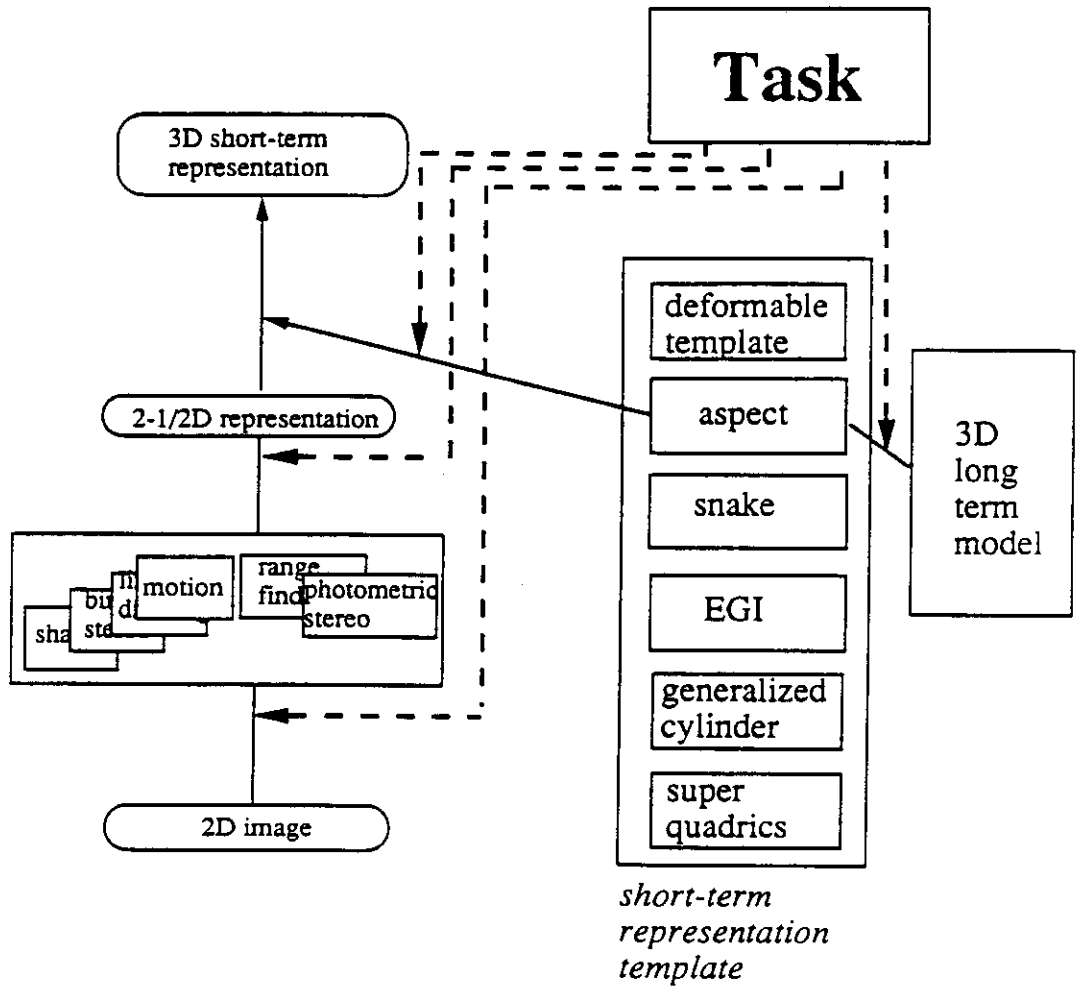


Figure 6: Task-oriented paradigm

- *Hook grasping* – pulls an object toward particular directions. The contact occurs at the points along the cross-sectional hemicircle.
- *Lateral grasping* – pushes an object on a soft side surface of one finger by the other fingers. The contact occurs at a point and points on a plane.
- *Palmar grasping* – grasps the end of bar by closing three fingers. The contact occurs at the three points.
- *Tip grasping* – grasps an object by closing two fingers from two opposite directions. We can achieve very fine grasping. The contact occurs at the two opposite points.

Spherical, cylindrical and hook graspings are grouped as *power* grasping family, while lateral, palmar and tip graspings are grouped as *precision* grasping family.

Once a grasping strategy is given, we have to choose one particular representation suitable to the strategy. Here, the issue is to investigate the relationship between required functional capabilities, representations, and grasping strategies.

Figure 8 summarizes the required functional capabilities for representations by these six grasping strategies. In the figure the sign “ ~ ” indicates approximated. Thus, the figure reads as that the spherical grasping requires approximated radius and approximated center of the object grasped.

We can summarize that the three power grasping strategies – spherical, cylindrical, and hook – require only approximated parameters rather than detailed parameters. For these grasping strategies, weak models such as superquadric representations are suitable for representing the object grasped.

The three precision grasping requires the existence of such detailed models as a geometric model. The lateral grasping requires the exact position of the two planes. The palmar grasping requires the knowledge of the exact position of the cross-section, while the tip grasping requires knowledge of the exact position of two contact points. In order to extract such exact information, we need an exact object model which is represented using polyhedrons. Since it is difficult to extract such information precisely in run time, we need a polyhedral approximation such as the one provided by a geometric model of the object.

The next important intermodule constraint is how to determine an appropriate group of feature for extracting such a representation. Let us focus on the case of precision graspings. For this grasping, the existence of a geometric model is a

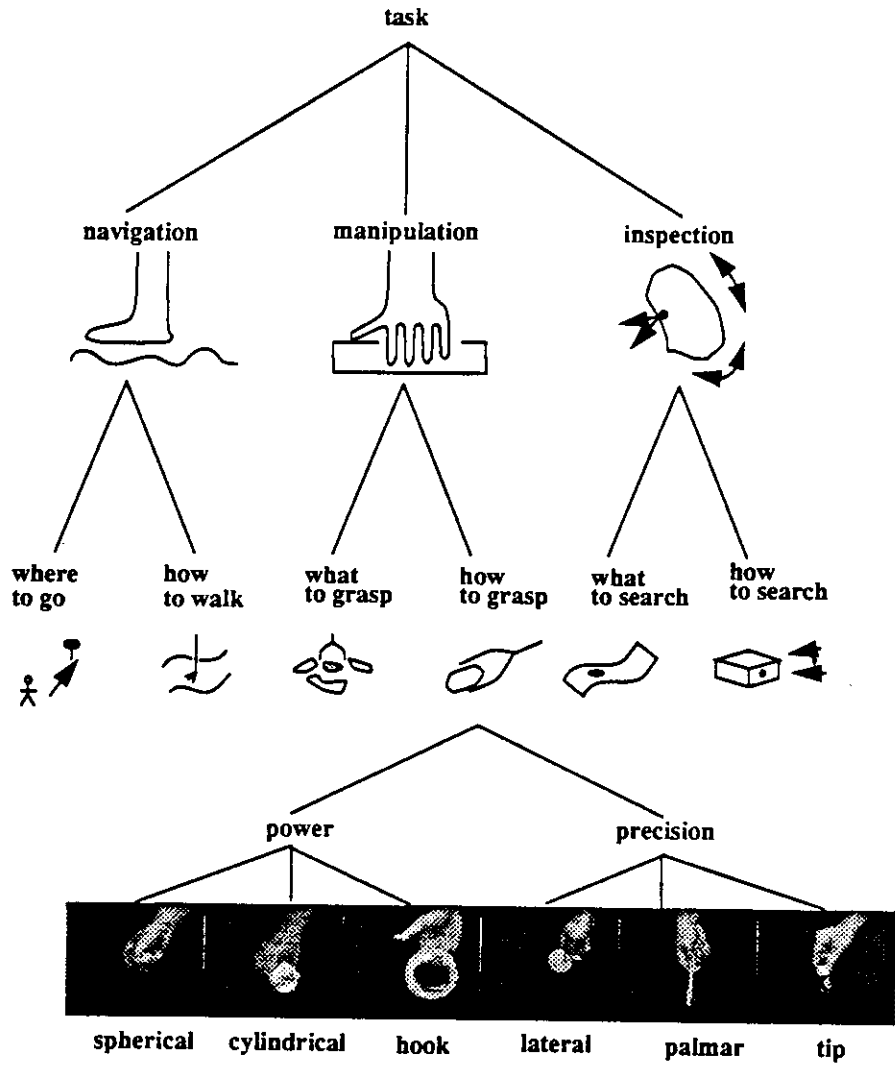


Figure 7: Taxonomy of task








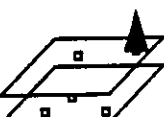
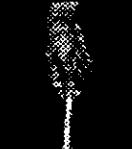



grasp strategy	required functional capabilities	representation
	 ~center ~radius	superquadrics
	 ~center ~radius ~axis direction	generalized cylinder
	 ~center ~radius ~axis direction ~pulling direction	superquadrics + pulling direction
	 orientation position of two planes width	two parallel planes (geometric model)
	 center radius	cross-sectional shape (geometric model)
	 position of points orientation	two contact positions (geometric model)

Figure 8: Required functional capabilities by grasping strategies

prerequisite. Then, the issue can be translated into the one how to select optimal features automatically from a geometric model. Goad proposes a method for selecting appropriate edges for object localization [9]. Bolles and Cain propose a focus feature method which selects important features and less important secondary features [6]. We have been developing a CMU vision algorithm compiler which chooses the optimal set of features for object localization [11].

For further segmentation and image acquisition, such issues as “what kind of sensor should be used”, “where it should be located to detect the necessary features”, and “what kind of features must be extracted” have been investigated under active sensing strategy generations [15, 8, 23, 10, 7].

By using Task-oriented vision paradigm as the design philosophy for vision systems architectures, we can unify recent active-vision [1, 4, 2, 12] and vision algorithm compiler [6, 23, 14] accomplishments toward the single unified goal, accomplishments toward completion of the task-oriented vision paradigm.

6 Conclusion

This paper presented a task-oriented vision approach to the design of vision systems. The task-oriented vision approach proposes to change the architecture of a vision system in a systematic fashion which depends on each task specification. We have presented a task oriented approach for systems that involve the localization and grasping of 3-D objects. In this case, the general methodology involves analyzing the task specification to derive the constraints on and requirements of the vision components. This starts with the type of representation, derived from the type of grasping selected, and continues down to the type of sensor. The task oriented approach is applicable to a wide range of vision systems. The task oriented approach will not only build more robust systems but will also give a new direction to vision research.

7 Acknowledgement

The authors wish to thank H. Delingette, T. Choi and Y. Yen for building some of the rock-sampling softwares, K.S. Hong and K.D. Gremban for building some of the bin-picking softwares. Takeo Kanade and Raj Reddy provided many useful comments and encouragements. Kathryn Porsche and Fredric Solomon proofread drafts of this paper and provided many useful comments which have improved the readability of this paper. The authors also thank Shree Nayar, Hideichi Sato, Yoshimasa Fujiwara and the members of the Task-oriented Vision Laboratory, the Robotics Institute, Carnegie Mellon University, for their valuable comments and suggestions.

References

- [1] J. Aloimonos. Purposive and qualitative active vision. In *Proceedings of DARPA Image Understanding Workshop*, pages 816–828, Pittsburgh, PA, 1990. Morgan Kaufmann.
- [2] R. Bajcsy, R. Paul, X. Yun, and V. Kumar. A multiagent system for intelligent material handling. In *Proceedings of 91 Intern. Conf. on Advanced Robot*, pages 18–23, Pisa, Italy, 1991. IEEE Computer Society.

- [3] R. Bajcsy and F. Solina. Three dimensional-object representation revisited. Technical Report MS-CIS-87-19, Univeristy of Pennsylvania, Department of Computer and Information Science, March 1987.
- [4] D. Ballard. Animate vision. In *Proceedings of Intern Conf on Artificial Intelligence*, 1989.
- [5] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. An autonomous rover for exploring mars. *IEEE Computer*, (6), June 1989.
- [6] R. Bolles and R. A. Cain. Recognizing and locating partially visible objects: the local-feature-focus method. *The International Journal on Robotics Research*, 1(3):57–82, 1982.
- [7] C.H. Chen and A.C. Kak. A robot vision system for recognizing 3-d objects in low-order polynomial time. *IEEE Trans. on Systems, Man, and Cybernetics*, 19(6):1535–1563, November/December 1989.
- [8] C.K. Cowan and A. Bergman. Determining the camera and light source location for visual task. In *IEEE Intern Conf Robotics and Automation*, pages 509–514, 1989.
- [9] C. Goad. Special purpose automatic programming for 3D model-based vision. In *Proc. of DARPA Image Understanding Workshop*, pages 94–104. DARPA, 1983.
- [10] C. Hansen and T. Henderson. CAGD-based computer vision. In *Proc. IEEE Computer Society Workshop on Computer Vision*, pages 100–105, Miami Beach, FL, December 1987. IEEE Computer Society.
- [11] K.S. Hong, K. Ikeuchi, and K.D. Gremban. Minimum cost aspect classification: a module of a vision algorithm compiler. In *10th Intern. Conf. on Pattern Recognition*, Atlantic City, N.J., June 1990. (a slightly longer version is available as CMU-CS-90-124).
- [12] S.A. Hutchinson and A.C. Kak. Planning sensing strategies in robot work cell with multi-sensor capabilities. *IEEE. Trans. Robotics and Automation*, 5(6):765–783, December 1989.

- [13] K. Ikeuchi. Determining a depth map using a dual photometric stereo. *The International Journal of Robotics Research*, 6(1):15–31, 1987.
- [14] K. Ikeuchi and K. S. Hong. Determining linear shape change: Toward automatic generation of object recognition programs. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 53(2), March 1991. a longer version, containing programs, is available as CMU-CS-88-188.
- [15] K. Ikeuchi and T. Kanade. Modeling sensors: Toward automatic generation of object recognition program. *Computer Vision, Graphics, and Image Processing*, (48):50–79, 1989.
- [16] K. Ikeuchi, H.K. Nishihara, B.K.P. Horn, P. Sobalvarro, and S. Nagata. Determining grasp points using photometric stereo and the prism binocular stereo system. *The International Journal of Robotics Research*, 5(1):46–65, 1986.
- [17] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Intern. Journal of Computer Vision*, 2(1):321–331, 1988.
- [18] D. Marr. *Vision*. Freeman, San Francisco, 1982.
- [19] A. P. Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28(2):293–331, 1986.
- [20] K. Sato, H. Yamamoto, and S. Inokuchi. Range imaging system utilizing nematic liquid crystal mask. In *International Conf. on Computer Vision*, pages 657–661, London, 1987.
- [21] F. Solina. Shape recovery and segmentation with deformable part model. Technical Report MS-CIS-87-111, University of Pennsylvania, Department of Computer and Information Science, December 1987.
- [22] C.L. Taylor and R.J. Schwarz. The anatomy and mechanics of the human hand. *Artificial Limbs*, 2:22–35, 1955.
- [23] S. Yi, R.M. Haralick, and L.G. Shapiro. Automatic sensor and light source positioning for machine vision. In *Proceedings of the 10th Intern. Conf. on Pattern Recognition*. IEEE Computer Society, 1990.