Proceedings of the 2002 IEEE
International Conference on Robotics & Automation
Washington, DC • May 2002

A Versatile Depalletizer of Boxes Based on Range Imagery

Dimitrios Katsoulas† Lothar Bergen†
Lambis Tassakos‡
†Computer Science Department, University of Freiburg, 79110 Freiburg, Germany
‡Inos Automationssoftware GmbH, 70563 Stuttgart, Germany

Abstract

We present a novel box depalletizing system based on images acquired with a time of flight laser sensor mounted on the hand of the robot. Scanning the upper layer of the pallet yields a 2.5D image to which edge detection and robust line fitting are applied to extract 3D vertices. This vertex information is used as an input for a model based object recognition system. Model vertices are matched to scene vertices and object location hypothesis are formed. These hypotheses are verified or rejected using a two-step verification process. Due to the fact that we use edges to extract vertices, rather than surfaces, we are able to detect target objects, in both cluttered and ordered configurations. Our experiments with different configurations of card board boxes and paper tissue packets demonstrate the validity of our approach. The main advantages of our system are it's versatility, simplicity, efficiency and robustness.

1 Introduction

This paper addresses the depalletizing problem (or robotic bin picking problem) in the context of which a number of objects of arbitrary dimensions, texture and type must be automatically located, grasped and transferred from a pallet (a rectangular platform), on which they reside, to a specific point defined by the user. The need for a robust and generic automated depalletizing system stems primarily from the car and food industries. An automated system for depalletizing is of great importance because it undertakes a task that is very monotonous, strenuous and sometimes quite dangerous for humans. In this contribution we deal with the construction of a depalletizer dealing with cluttered and ordered configurations of boxes.

1.1 Related Work

Existing systems can be classified as follows: systems incorporating no vision at all and systems incorporating vision. The majority of the systems employed in industrial depalletizing applications so far do not contain any vision modules. They usually employ preprogrammed gantry robots for bulk depalletizing tasks. Vision systems can be classified in systems using intensity and range imagery. The former have all problems of camera-based object identification: sensitivity to lighting conditions and target object texture.

Attempts at incorporating range imagery seem much more promising. In [1] a structured light range sensor was employed to deal with unloading piles of postal parcels. Due to the usage of intensity based feature extraction methods on range images, the image features are not detected with high accuracy. In the event that no objects are detected, the robot is commanded to disturb the pallet and therefore target objects risk being damaged. The system of [2] uses a laser sensor to deal effectively with neatly placed boxes. The system, although fast and accurate, cannot deal with cluttered configurations. In [3] the authors deal with depalletizing of parallelepipeds of unknown dimensions with the help of a range sensor. Complete scene understanding is attempted. They use methods which facilitate size and pose estimation of target objects by virtually extending their dimensions in the direction away from the sensor until they physically contact other objects in the scene. Efficiency measurements have not been presented by the authors.

One of the fastest and most elegant recognition systems for 3D convex objects, developed so far, is described in [4]. In this system, immediately related model features are grouped into sets, which are named local feature sets (LFSs). The matching of a model LFS with a scene feature group assumes the existence of the model in the pile and generates a unique value for the pose transform which takes the particular model object to the scene. This model location hypothesis is then rejected or verified by checking if the position of neighboring scene feature groups match positions of other model LFS. The authors have chosen to use 3D vertices as LFSs, since they provide constraints for calculating the pose transform.

The accurate calculation of the vertex position is of extreme importance for the accuracy of recognition. The authors employ a region based range image segmentation technique and the vertex position is determined by intersecting surfaces. In the event that the objects expose three surfaces to the sensor, the calculation of the vertex point is as accurate as desired. The problem is that in many object configurations this is not the case. If only two surfaces of the object are exposed, no reli-

0-7803-7272-7/02/$17.00 © 2002 IEEE

able method for computing the coordinates of the vertex points is proposed in [5]. What is more, in configurations where objects expose only one surface, the calculation of vertex points based on surface information is impossible.

1.2 Our Approach

In our system a time of flight laser sensor mounted on the robot's hand is used for acquiring images. The feature detection is based on edges. We employ a fast range image edge detection algorithm to acquire the image's edge map. The object edges are robustly calculated. Vertex points are computed as intersection of two edges. LFSs in our system consist of a vertex point and the two edge directions. The verification criterion handles cases where few box features are detected. In this way we are able to recognize objects even if only one surface is exposed to the laser source.

Our system demonstrates a plethora of advantages: Computational efficiency, due to the fast feature extraction and object recognition subsystems it incorporates. Accuracy, due to the fact that robust feature estimators are utilized. Robustness, since a two step verification strategy minimizes the number of false detections. Ease of installation, since a simple mounting of the laser sensor on the hand of the robot is all we need. Lighting condition and target object texture independence, since we use data acquired from a time of flight laser sensor. Versatility, since our system deals in the same way with cluttered and neat configurations of boxes. And last but not least simplicity, as the flow diagrams of the systems components which follow demonstrate.

In the following paragraphs our system is described in detail. An experimental results paragraph illustrates the validity of our approach.

2 System Description

From the hardware point of view, our system comprises an industrial robot (model KR 15/2 manufactured by KUKA GmbH), a square vacuum-gripper, which grasps the boxes from one of the surfaces they expose, and a time of flight laser sensor (model LMS200, manufactured by SICK GmbH). The sensor is integrated on the gripper and the latter is seamlessly attached to the robot's flange. In this way, we take full advantage of the flexibility for viewpoint selection provided by a six degree of freedom robot.

The operation of our system is depicted in Fig. 1. The robot is programmed to execute a linear scanning movement, the end points of which are the mid points of the two opposite sites of the rectangular pallet. In this way the upper layer of the pallet is scanned and a range image is acquired (Fig. 1(a) and (b)). The object recognition systems accepts the range image and recognizes the boxes. The coordinates of the detected objects are sent to the robot which then grasps them (Fig 1(c) and (d)). In the paragraphs that follow, the feature extraction box recognition algorithms of the robotic system are described in detail.

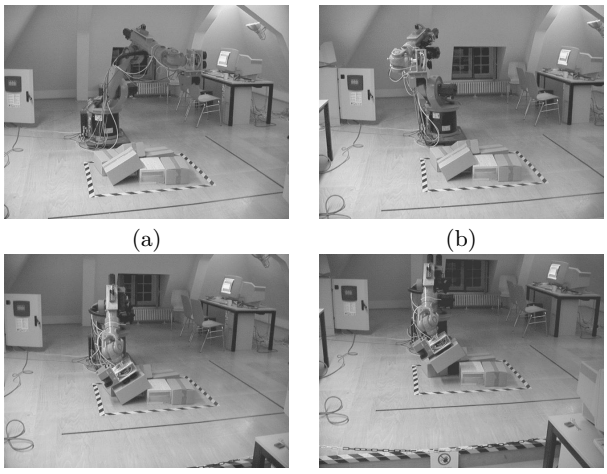


Figure 1: Operation example

2.1 Data acquisition

In the scanning phase of the system (Fig 1(a) and (b)), the scanning plane of the sensor is perpendicular to the direction of the movement and the sensor faces the upper side of the pallet. During this phase, a set of 2D laser scan lines is acquired. This set of scan lines yield a 2.5D image of the scene. What is important for what follows is that the image coordinates of each three dimensional image points are known. Finally, the noise introduced through range and reflectance changes is dealt with using the noise attenuation method described in [6]. The acquisition time for the retrieval of a scan line of the sensor is 25 ms. Since the length of our rectangular platform is about 1.5 meters and since we want the distance between scan lines not to be more than 1cm, the duration of the data acquisition process is about 4 seconds.

2.2 Edge Detection

Edges are detected by applying a scan line approximation based edge detector to the rows and columns of the 2.5D image. A detailed description of the algorithm can be found in [7]. Due to the fact that the algorithm uses information from long line segments to compute the edge points, it has an advantage over local approaches in terms of accuracy. A discussion of the accuracy and the computational complexity can be found in [2].

2.3 3D Vertex Detection

The vertex detection consists of two steps: robust fitting of line segments and the identification of the line segments which are close to one-another and which roughly form a

90 degree angle.

When thinking of robust line fitting, the Hough transform (HT) springs to one's mind. Unfortunately, the computational complexity and the memory requirements, which rise exponentially with the number of parameters under detection, make the HT impractical for our problem. The solution to this problem lies in the fact that a large number of connected components (CC) can be found in the edge data, which can be used to obtain a coarse segmentation: points lying in the vicinity of lines fitted to the CC in the least square sense can be used as input for a subsequent robust line fitting algorithm. For the robust fitting on of the standard techniques, such as Least Median of Squares or RANSAC can be used. However, for the sake of simplicity and computational efficiency, we have developed a two stage line fitting algorithm which has been inspired by the one-dimensional accumulation used in the dynamic generalized Hough transform [8] and which will be described in the following.

The first step consists of the determination of the direction $\mathbf{d}$ of the line. Therefore, pairs of points are sampled, their difference vectors normalized and the components of the normalized vectors accumulated in three one-dimensional accumulators. The maxima of the accumulators yield the components of the direction vector $\mathbf{d}$. In the second step, the difference vectors between the line through the origin with the direction vector $\mathbf{d}$ and the data points are determined and once again accumulated in three one-dimensional accumulators. The maxima of the accumulators then yield a vector $\mathbf{p}$ which indicates the position of a point on the line with respect to the origin. Of course, the parameterization with its 6, rather then the 4 necessary parameters, is redundant, but has the advantage of not requiring any coordinate transforms. Figure 2 shows the results of our robust fitting technique for two segments: each CC is represented with crosses, the edge points with dots, the result of the least square fitting with a thin line and the final result with a thick line.

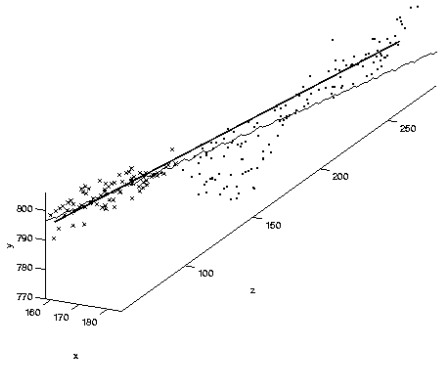


Figure 2: 3D-line fitting

In order to obtain vertices, the pairwise distances of the line segments are calculated and lines which are close to one-another and form roughly a 90 degree angle are grouped into a scene vertex. The intersection of the two lines is defined as the midpoint between the lines. We represent a scene vertex as a triplet $(\mathbf{V_S}, \hat{\mathbf{V}}^{\mathbf{1}}_{\mathbf{S}}, \hat{\mathbf{V}}^{\mathbf{2}}_{\mathbf{S}})$, where $\mathbf{V_S}$ is the position of the vertex point in the scene, while $\hat{\mathbf{V}}^{\mathbf{1}}_{\mathbf{S}}$ and $\hat{\mathbf{V}}^{\mathbf{2}}_{\mathbf{S}}$ are the normalized vectors pointing in the direction of the edges joining at the vertex. In the event that the distance of two scene vertices is below a certain threshold, the vertices are considered to belong to the same scene vertex and corresponding elements of the triplets are averaged to obtain the final values.

2.4 Object Recognition

Our object recognition system is based on a hypothesis generation and verification framework. In a data driven approach, a scene vertex is matched to a vertex of one of the models stored in the model database. This match creates a hypothesis about the position of the model in the scene. Whether the hypothesis will be verified or rejected depends on the position and the orientation of neighboring scene vertices.

A Local Feature Set (LFS) is defined to be a set of features of a model box, a 3D vertex in our case, which, when matched to a scene vertex, produces a unique pose transform that takes a model object to the scene. This pose transform is equivalent to a hypothesis of the model box location in space. We denote this transform as ${}^{S}_{M}T$, since it describes the position of the model ($\{\mathbf{M}\}$) relative to the scene coordinate system ($\{\mathbf{S}\}$). The center of gravity of all models resides at the origin of the model coordinate frame (c.f. Fig. 3).

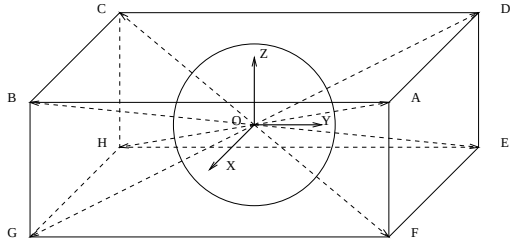


Figure 3: Model box

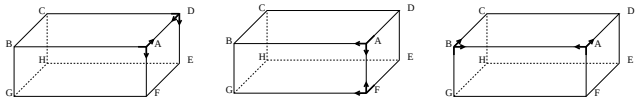


Figure 4: Local feature sets (LFSs)

In the same way as the scene vertices, the LFSs are represented as $(\mathbf{V_M}, \hat{\mathbf{V}}^{\mathbf{1}}_{\mathbf{M}}, \hat{\mathbf{V}}^{\mathbf{2}}_{\mathbf{M}})$, where $\mathbf{V_M}$ is a model vertex position vector in the model coordinate system, while $\hat{\mathbf{V}}^{\mathbf{1}}_{\mathbf{M}}$ and $\hat{\mathbf{V}}^{\mathbf{2}}_{\mathbf{M}}$ are the normalized directions of the model edge

vectors joining at the vertex $\mathbf{V_M}$. Note that in this representation the order of the direction vectors is such that their cross product points in the same direction as the third edge of the box corresponding to the vertex. Note as well the edge direction vector pair defines a model surface. Since a box has three different surfaces, and since for each surface two orientations in space are valid, the number of LFSs per model should be six. A list of all the LFSs of all the box models in our application are stored in the model database. For the model box of the Fig. 3, the LFSs are shown in the Fig. 4 and listed in Table 1. The verification of a hypothesis generated by a model

LFS ID	1	2	3	4	5	6
Vertex	**A**	**D**	**A**	**F**	**A**	**B**
Direction vector 1	$\hat{\mathbf{AD}}$	$\hat{\mathbf{DE}}$	$\hat{\mathbf{AB}}$	$\hat{\mathbf{FG}}$	$\hat{\mathbf{AB}}$	$\hat{\mathbf{BC}}$
Direction vector 2	$\hat{\mathbf{AF}}$	$\hat{\mathbf{DA}}$	$\hat{\mathbf{AF}}$	$\hat{\mathbf{FA}}$	$\hat{\mathbf{AD}}$	$\hat{\mathbf{BA}}$

Table 1: LFS List

LFS and expressed by the transform ${}^S_M T$ is aided by the feature sphere. The feature sphere is a data structure in the model database, which allows for rapid verification of a pose transform calculated by matching a scene vertex to an LFS. Conceptually, the feature sphere is a 3D sphere whose radius equals unity, centered at the origin of the model coordinate system (Fig. 3). Contents of the feature sphere are the normalized position vectors of the 8 vertices of our model box (principal directions of vertex points). For a detailed description of the implementation the reader is referred to [4].

The flow diagram of the object recognition system is depicted in Fig. 5. Input of our system is the list of detected scene vertices, the scene feature sets. An element is extracted from the list and all the adjacent vertices of the feature set are retrieved. Two scene vertices are considered to be adjacent when their distances is smaller than the diagonal of the largest model box in the model database. A model LFS is then extracted from the model LFS list. A matching between the model LFS and the scene vertex is performed and a pose location hypothesis is generated. If the hypothesis is verified, the elements of the adjacent scene vertex list which are compatible are removed form the list of scene vertices. Additionally the transform corresponding to the verified hypothesis is added to the list of verified transforms. This list will be forwarded to the robotic grasping subsystem to grasp the boxes. If the hypothesis is not verified, on the other hand, the next model LFS is extracted. If all the LFSs are examined, another scene vertex is extracted from the list and so on until all scene vertices are examined.

In the paragraphs that follow, the salient procedures of the object recognition algorithm, that is the hypothesis generation and verification, are described in detail.

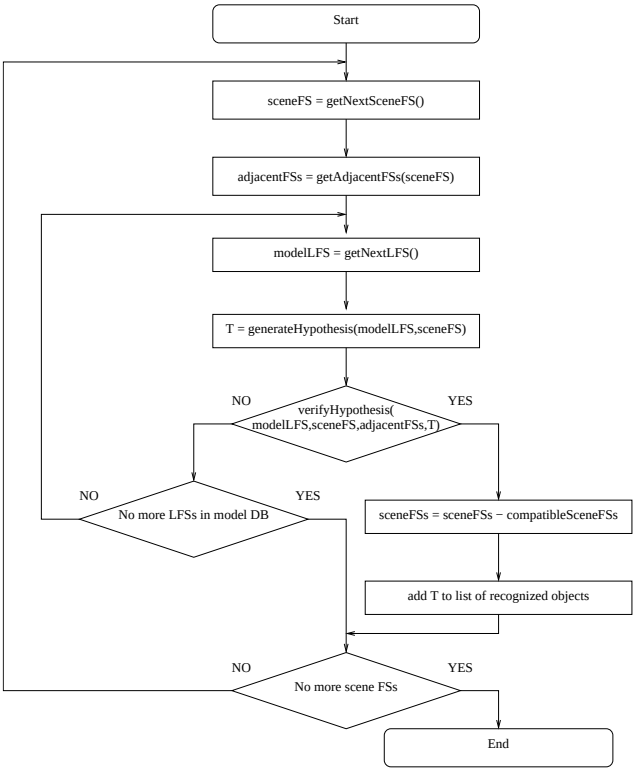


Figure 5: Object recognition flow diagram

2.4.1 Hypothesis generation: An LFS generates a unique box location hypothesis in the scene. The hypothesis is expressed by a transform ${}^S_M T$. If $(\mathbf{V_S}, \hat{\mathbf{V}}^{\mathbf{1}}_{\mathbf{S}}, \hat{\mathbf{V}}^{\mathbf{2}}_{\mathbf{S}})$ a scene vertex and $(\mathbf{V_M}, \hat{\mathbf{V}}^{\mathbf{1}}_{\mathbf{M}}, \hat{\mathbf{V}}^{\mathbf{2}}_{\mathbf{M}})$ a model LFS, the transform is calculated by matching these two entities. First, the rotation matrix $\mathbf{R}$ is computed by minimizing the matching error between the two edge direction vectors:

$$E^2 = \sum_{i=1}^{2} \|\mathbf{R}\hat{\mathbf{V}}^{\mathbf{i}}_{\mathbf{M}} - \hat{\mathbf{V}}^{\mathbf{i}}_{\mathbf{S}}\|^{\mathbf{2}}. \quad (1)$$

Then, we use the vertex position vectors $\mathbf{V_M}$ and $\mathbf{V_S}$ to compute the translation vector:

$$\mathbf{t} = \mathbf{V_S} - \mathbf{R}\mathbf{V_M}. \quad (2)$$

An elegant solution for this minimization problem, which is based on quaternions, can be found in [9].

2.4.2 Hypothesis verification: The hypothesis verification procedure determines whether an hypothesis given by the transform ${}^S_M T$ is valid. Primary target in the design of the verification procedure is the minimization of the number of false positives, which guarantees robust performance of the system. Input of the hypothesis verification procedure is a list of adjacent vertices of the scene vertex which determined the transform ${}^S_M T$. The vertex

positions are then back-transformed to the model frame. If $\mathbf{V_S^a}$ is an adjacent vertex position, then the position in the model frame is computed in the following manner:

$$\mathbf{V_M^a} = {}_{\mathbf{S}}^{\mathbf{M}}\mathbf{T}\mathbf{V_S^a} = \mathbf{R^{-1}}(\mathbf{V_S^a} - \mathbf{t}). \quad (3)$$

The direction of the position vector $\mathbf{V_M^a}$ is then normalized. If one of the principal directions of the model stored in the feature sphere is close to the input direction vector, we consider the adjacent scene vertex **compatible** with the examined scene vertex under the current hypothesis. Due to the way the feature sphere is implemented, searching for a stored direction is a rapid process.

Whether a hypothesis will be verified or not depends on the number of compatible adjacent scene vertices. The detection of two compatible vertices, each sharing no common edges with all the others, is required to safely recognize a box in three dimensions. However, in many cases, like when the boxes are neatly placed on the platform, the boxes expose less than two surfaces and therefore the verification criterion cannot be satisfied. In these cases we loosen the set of constraints of the criterion, so as to verify a rectangular surface hypothesis, rather than a 3D box position hypothesis. The disadvantage is that if two model boxes have a surface with equal dimensions, the verification procedure will assert that the particular surface was detected, but will be not able to distinguish which of the two boxes reside on the pile. If all the surfaces of our models have different sizes, this is equivalent to a verification of a box hypothesis.

A minimum set of requirements for surface verification is that the number of compatible adjacent vertices is two. However, if only one compatible vertex is detected, the verification framework can still accept a surface pose hypothesis if the compatible vertex shares no common edge with the scene vertex on which the hypothesis generating LFS was matched. If however this does not happen (that is when the compatible adjacent vertex shares an edge with the scene vertex) two hypotheses about the position of the box, to which the surface belongs exist. Using the feature sphere for verification here is not enough for determining the box position. For this reason we introduce an additional test in the verification process. We extract the 3D points lying in the hypothesized surface segment, fit a 3D plane to the points, and if the fitting error is below a certain threshold we consider the surface segment verified. The identification of the inlying points is straightforward: The four points defining the boundary of the hypothesized surface are computed and their image coordinates extracted. The image points inside the image polygon defined by the four boundary points are determined with a very fast polygon rasterization algorithm [10].

The Flow diagram of the overall verification process is depicted in Fig. 6. For robust verification with the fitting test, we have to be sure that the hypotheses corresponding to surfaces with bigger area are tested before those corresponding to surfaces with smaller area. For this reason, the model LFSs are placed in the LFS list of the model database in descending order of the area of the surface to which they correspond.

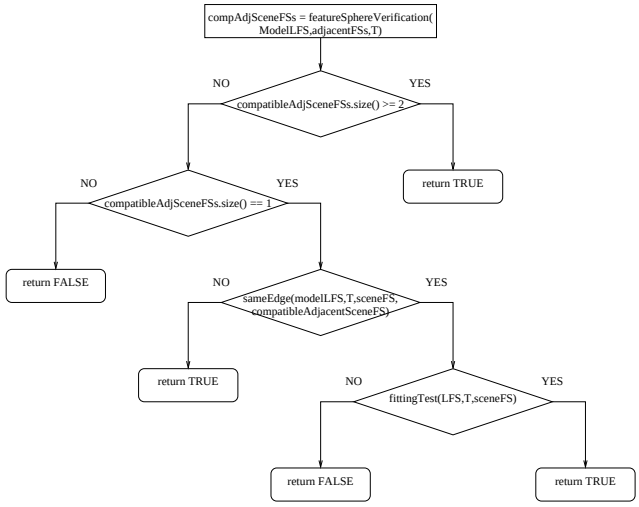


Figure 6: Hypothesis verification flow diagram

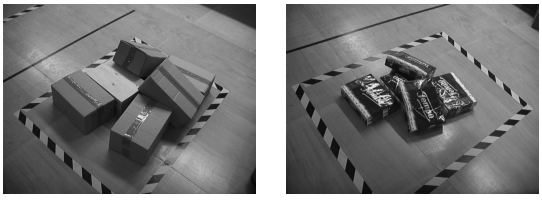
Note that the plane fitting test could be as well used as a verification method in the event that no compatible scene vertices were detected. For a particular scene vertex the fitting test would then be triggered a number of times equal to the number of model LFSs, in the worst case. Although our plane fitting test is fast, the time overhead that such an approach would introduce made us avoid its usage in such cases. In addition, in almost all cases two vertices corresponding to the same box can be detected.

The introduction of the plane fitting test in the verification procedure has minimized the number of scene vertices that need to be detected. Furthermore, this test ensures that a detected box surface is graspable (i.e. that there are no other objects lying on the top of it). This could not have been determined if the four points defining the boundary of the surface were not accurately computed. The accuracy in the calculation of these points stems from the accurate computation of the points and the edges of the scene vertices.

Additionally, a successful fitting test implies that almost all 3D points lie inside the boundary and thus belong to a graspable surface. The parameters of the fitted plane can therefore be used for even more robust grasping of the box to which the surface corresponds.

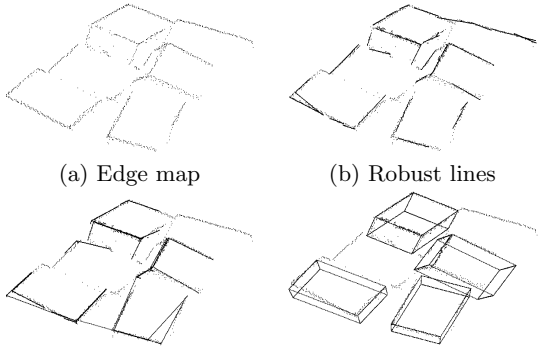
3 Experimental Results

We tested the system by using two kinds of boxes. Card board boxes and paper tissue packets (box - like) wrapped in a foil. The dimensions of the card boxes where 35cm in length, 23cm in width and 15cm in height. The dimensions of the box like objects where 27.3, 21.5 and 7cm respectively. The pallet is a rectangular area, 1.5m in length and about 0.7m in width.



(a) Card board boxes (b) Paper tissue packets

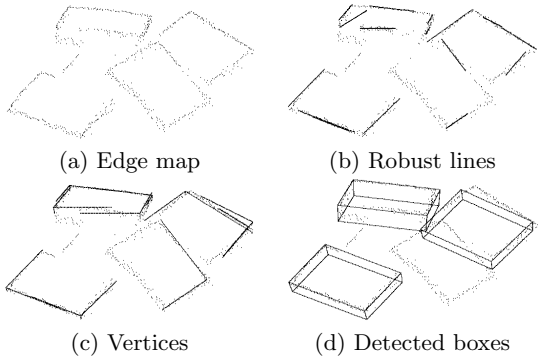
Figure 7: Object configurations



(a) Edge map (b) Robust lines

(c) Vertices (d) Detected boxes

Figure 8: Card board box recognition



(a) Edge map (b) Robust lines

(c) Vertices (d) Detected boxes

Figure 9: Paper tissue packet recognition

Two example configurations of objects, are depicted in Fig 7(a) and (b). Note that in these configurations both jumbled and neatly placed objects exist. The results of our recognition algorithms (edge detection, robust box edge estimation, vertex detection and object recognition) on the two configurations are depicted in Fig. 8 and 9 respectively. The reader can see in these figures that the detection accuracy allows for a precise object grasping. This was our observation after having let the system run and grasp several objects. Nevertheless, we have not yet performed detailed accuracy measurements of the system. This is something we plan to do in the near future.

For both configurations the execution time of our algorithms was about 11 seconds on a Pentium III 650 MHz PC. If we add to this the time needed for the data acquisition (about 4s), we see that the overall processing time is about 15 seconds. Note that in both example configurations, more than one objects were detected. Due to this fact, the average box localization time is considerably less. In terms of robustness, our experiments demonstrated that the system only occasionally fails to find at least one graspable box in the pile.

In the event of neatly placed objects, where the boxes are placed very close together in distinct layers, however, our system failed to recognize the boxes, since the resolution of the sensor didn't allow the edges between the objects to be detected. In such cases an additional sensor like a normal intensity camera should be employed to determine the orientation of the objects. Such an approach is described in [2].

4 Conclusions and Future Work

A box depalletizing system based on range images acquired with a time of flight laser sensor has been presented. We have shown how the use of edge detection and robust line fitting allows for precise 3D vertices detection. Using these vertices in a hypothesis generation and verification framework makes object recognition both robust and accurate. Due to the fact that edges, rather than surfaces, are employed for the vertex detection, our system is able to deal with objects even if they expose less than three surfaces to the sensor. Thus the system is capable of recognizing boxes in cluttered and ordered configurations. However, the system cannot resolve situation where the boxes a neatly placed in layers and contact one-another so that edges between them cannot be detected. In the future we will perform detailed grasping accuracy measurements. Furthermore, we plan to exploit the vertices detected in previous scans to make the recognition process faster.

5 Acknowledgements

This work was supported for the first author by the German ministry of economy and technology under the

PRO INNO program, Grant No. KF 0185401KLF0.

References

[1] A.J. Baerveldt, *Robust Singulation of Parcels with a Robot System using multiple sensors*, Ph.D. thesis, Swiss federal institute of technology, 1993.

[2] D.K. Katsoulas and D.I. Kosmopoulos, "An efficient depalletizing system based on 2D range imagery," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-2001), Seoul, Korea*. May 21–26 2001, pp. 305–312, IEEE Robotics and Automation Society.

[3] A. J. Vayda and A. C. Kak, "A robot vision system for recognition of generic shaped objects," *Computer Vision, Graphics, and Image Processing. Image Understanding*, vol. 54, no. 1, pp. 1–46, July 1991.

[4] C. H. Chen and A. C. Kak, "A robot vision system for recognizing 3-D objects in low-order polynomial time," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 6, pp. 1535–1563, Nov.-Dec. 1989.

[5] C.H. Chen and A.C. Kak, "3D-POLY: A robot vision system for recognized objects in occluded environments," Tech. Rep. 88-48, Purdue University, 1988.

[6] M. D. Adams, "Amplitude modulated optical range data analysis in mobile robotics," in *Proceedings of the 1993 IEEE International Conference on Robotics and Automation: Volume 2*, Lisa Werner, Robert; O'Conner, Ed., Atlanta, GE, May 1993, pp. 8–13, IEEE Computer Society Press.

[7] D.K. Katsoulas and L. Bergen, "Efficient 3d vertex detection in range images acquired with a laser sensor," in *Pattern Recognition, Proc. of 23rd DAGM Symposium*, B. Radig and S. Florczyk, Eds. Sept. 2001, number 2191 in LNCS Pattern Recognition, pp. 116–123, Springer.

[8] V. F. Leavers, *Shape Detection in Computer Vision Using the Hough Transform*, Springer-Verlag, London, 1992.

[9] J. Andrade-Cetto and A.C. Kak, "Object recognition," in *Wiley Encyclopedia of Electrical and Electronics Engineering*, J. G. Webster, Ed., pp. 449–470. John Wiley & Sons, 2000.

[10] R. W. Swanson and L. J. Thayer, "A fast shaded-polygon renderer," in *Computer Graphics (SIGGRAPH '86 Proceedings)*, David C. Evans and Rusell J. Athay, Eds., Aug. 1986, vol. 20, pp. 95–101.