BIN OF PARTS STRATEGY

J. L. Turney

T. N. Mudge

May 1982

Center for Robotics and Integrated Manufacturing

Robot System Division

COLLEGE OF ENGINEERING
THE UNIVERSITY OF MICHIGAN
ANN ARBOR, MICHIGAN 48109

1. INTRODUCTION

The following is an outline of a possible strategy for segmentation, recognition, and acquisition of parts from a bin. This problem will be referred to as the Bin of Parts or BoP problem. It is assumed that the parts are of distinct types, and that the desired target part may be occluded and/or physically covered by non-target parts.

It is also assumed that the parts are not to be sorted, palletized, back-lit or placed in any sort of desirable environment such as a jig If any of the above strategies is adopted, it is understood that techniques using complete silhouette such as the Stanford Vision Module (which uses moment and topological descriptors), Fourier Descriptors (which needs an unbroken boundary as a periodic function), or the Cytocomputer [1] (using image algebra again on an unbroken boundary) are all adequate for the task and perhaps preferable (and in the order listed) to the following suggested strategy.

2. STRATEGY

The following strategy is based on the premise that when one looks through the bin, one need not recognize all objects in the field of vision, but only the desired target part, since the knowledge gained from an initial recognition of the entire bin is lost once the target part is removed (disturbing other parts in the process) or when new parts are added to the bin. It is, therefore, more important to search the bin only for locations and orientations of the desired part. Once this premise has been accepted the following composite of techniques is a reasonable approach.

---

[1] The Cytocomputer is a special purpose processing computer. "Cyctocomputer" is a registered trademark if the Environmental Research Institute of Michigan.

2.1. Edge detection

The initial image is replaced by its edges using Fry and Chen (moderately fast), Sobel (fast), Davies (slower but incorporates texture information), or relaxation techniques to pull edge points above threshold. For industrial parts, it is assumed edge information will provide the majority of the information about the part. Edge detection should be fast. Use TI's PIPE processor. See Fig. 1 for edge detectors.

2.2. Depth information using hierarchical search techniques

A coarse to fine (hierarchical) search strategy is adopted in matching scenes from one camera to another camera to determine depth by correlation. The image is converted to edges and its resolution is decreased by "shrinking down" its size using low pass spatial filtering (one method is to average neighboring pixels) followed by resampling of the image at a larger pixel separation. The window is similarly converted to edges and shrunk, and a search is instituted based on this reduced image and window. Spots of interest are noted it a check map (See Fig. 2) and are searched again at higher image resolution. (Sequential rules for searching are in Hall [2] p.485) This represents a log order search. If necessary, structured lighting, such as laser dots or scans, might be employed to obtain speed.

Depth information is seen as vital in correcting edge distortion due to tilt, completing broken edges and eliminating noise by using depth differentials between sides of questionable edges as the deciding factor, and as providing the robot with the distance and unobstructed approaches to the target part.

---

[2] *Computer Image Processing and Recognition*, E. L. Hall, Academic Press, 1979.

<u>Sobel</u> $\quad S(m,n) = (d_x^2 + d_y^2)^{1/2}$

$d_x = (f_{m-1,n-1} + 2f_{m,n-1} + f_{m+1,n-1}) - (f_{m-1,n+1} + 2f_{m,n+1} + f_{m+1,n+1})$

$d_y = (f_{m+1,n-1} + 2f_{m+1,n} + f_{m+1,n+1}) - (f_{m-1,n-1} + 2f_{m-1,n} - f_{m-1,n+1})$

<u>Frei + Chen</u>

$P_e = \left[\sum_{i=1}^{4} (w_i^t x)^2\right]^{1/2}$ where x is a 3×3 region concatenated into a vector and w_i for edges are

$w_1 = (1, \sqrt{2}, 1, 0, 0, 0, -1, -\sqrt{2}, -1)$

$w_2 = (1, 0, -1, \sqrt{2}, 0, -\sqrt{2}, 1, 0, -1)$

$w_3 = (0, -1, \sqrt{2}, 1, 0, -1, -\sqrt{2}, 1, 0)$

$w_4 = (\sqrt{2}, -1, 0, -1, 0, 1, 0, 1, -\sqrt{2})$

<u>Davies</u>

$S_{12} = |\sigma_1 - \sigma_2|$ where $\sigma_1 = |f_{m,n-1} - \mu| + |f_{m+1,n-1} - \mu| + |f_{m+1,n} - \mu|$

$\sigma_2 = |f_{m,n+1} - \mu| + |f_{m-1,n+1} - \mu| + |f_{m-1,n} - \mu|$ and

$\mu = \frac{1}{3}(f_{m,n-1} + f_{m+1,n-1} + f_{m+1,n})$

Figure 1. Edge detectors.

reduce image

reduce window

W_2 ⬛ — ◼ W_1

where window matches *image* *check map*

I_2 [◼] [x ←]

×
× ×

× ×
× ×

W_1 ◼

I_1

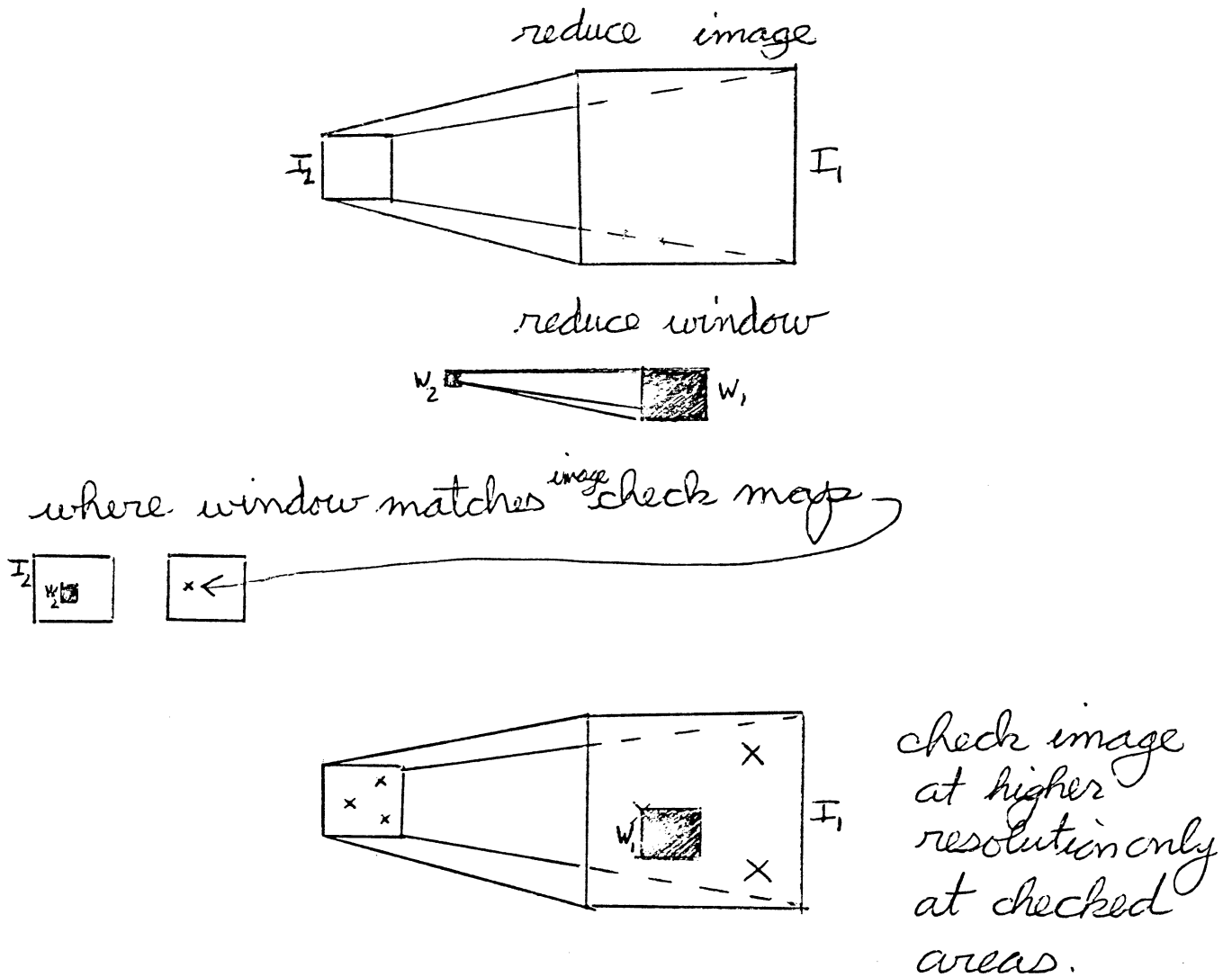*check image at higher resolution only at checked areas.*

Figure 2. Hierarchical search.

The depth algorithm is a special candidate for being off-loaded on a special purpose processor, since depth maps have a wider utility than in just their application to the BoP problem.

2.3. Determine Likely Edges for Template Fit

The longer edges are tested first. After they are tilt corrected, a theta-s plot is made on the edge, where theta is the tangent angle to the curve modulo two pi and s is the arc length along the edge. The function theta of s is then correlated with a similar function recorded for the target part in a library (See Perkins [3]).

A hierarchical approach is useful here where the shrunk edges are fitted first and then resulting edges of interest are investigated at higher resolution.

2.4. Library of Templates stored as Fourier Descriptors

Templates of the parts are stored as FD's. FD's are useful because they allow easy rotation, translation, scaling, tilting, interpolation between different viewing angles, and data reduction.

A hierarchical approach is desirable here also.

The FD is converted to real space and then matched against the curve under investigation. Correlation using pairing functions are used to achieve greater distinction between a match and the background and also for speed. A correlated point is counted only if neighbor points also correlate, otherwise it is rejected. See Fig. 4.

3. POSSIBLE ARCHITECTURE FOR VISION PROCESSOR

A set of processors with an asynchronous crossbar connection to a set of memories presents itself as a reasonable choice for the BoP problem since work such as the fine to coarse image reduction, depth determination, edge

---

[3] "A Model-Based Vision System for Industrial Parts", W. A. Perkins, *IEEE Transactions on Computers*, Vol. C-27, No. 2, Feb. 1978
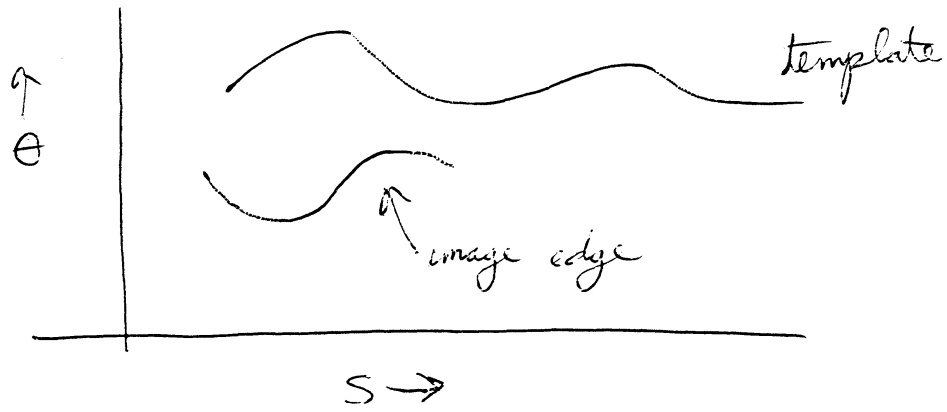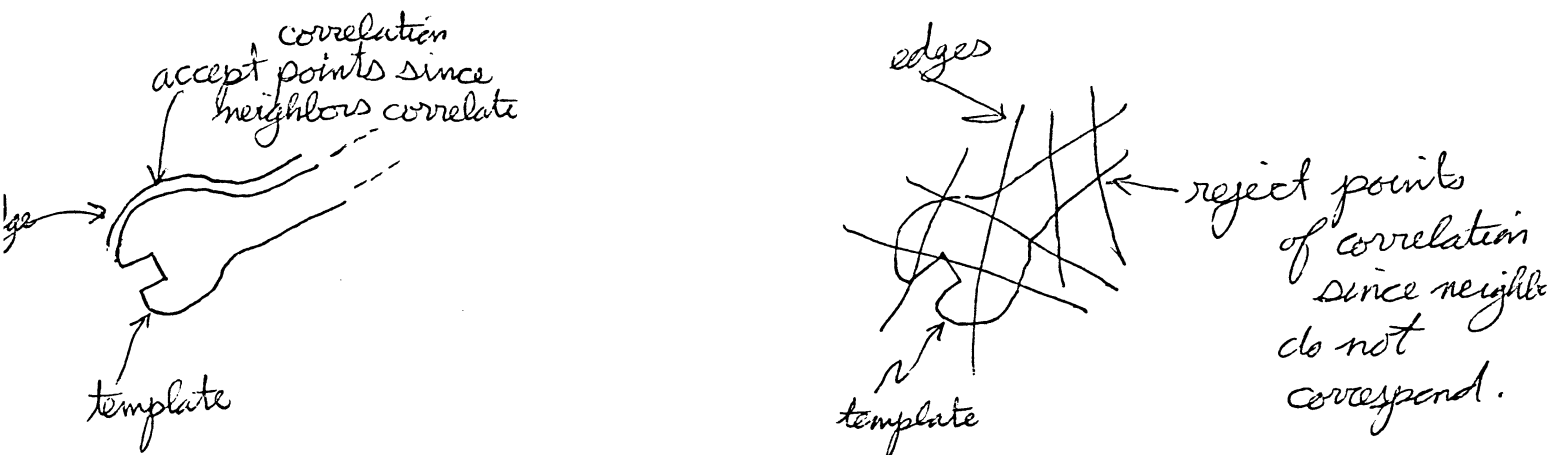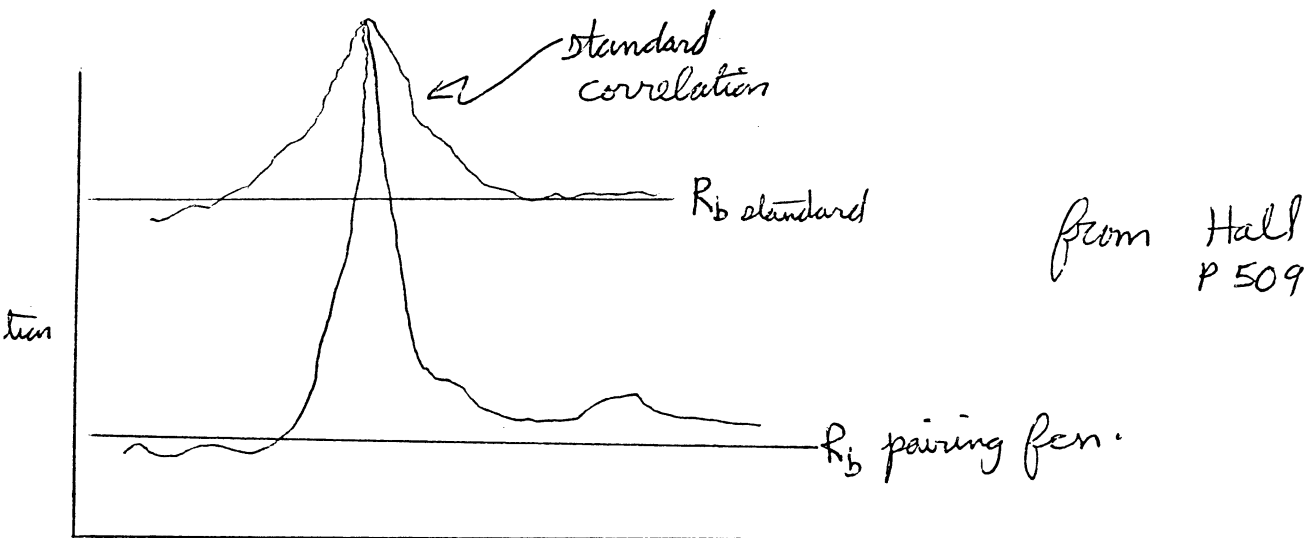
Figure 3. Theta-s matching.

Figure 4. Pairing function correlation vs. standard correlation and correlation rejection policy.

detection, regional radii histogramming, can be partioned easily (and represents many memory accesses for each crosspoint connection cycle). Also tasks such as the conversion from FD to 2D template followed be correlation of the template with the image can be roughly "pipelined". (See Fig. 5.

asynchronous
x bar
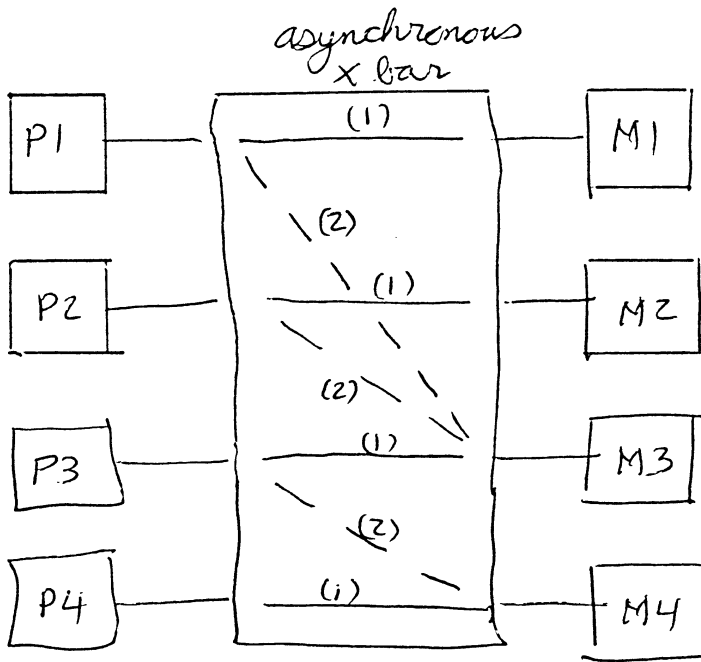
In mode (1) all processors operate on partitioned image. In other parts of program P1 & P2 operate on FD while P3 correlates and P4 performs another task. Information is passed during mode (2), P1 & P2 convert another rotation of FD to 2D template while P3 correlates.

Figure 5. Possible architecture in partitioned and "pipelined" mode.

4. SUMMARY

The key to the above strategy is the computational power to implement a depth search and template matches in times less than those use to, say, palletize parts. It would seem that there is a need for considerable computational power. Also all stages of the problems seem to be partitionable for multiprocessing applications.

Appendix A list an outline of the strategy. Appendix B list the programs needed and their attributes.

5. APPENDIX A. OUTLINE OF STRATEGY

Bin of Parts Strategy

I. Reduce image to successive levels of lower resolution

    A. Low pass filter

        1. Use simple averaging

        2. Use masks in Hall (See page 494)

    B. Resample filtered image

II. Reduce Images at all levels to edges

    A. Use template edge detectors (Frei and Chen), edge detectors with direction (gradient or Huekel), or nonlinear (Sobel, Davies has texture).

    B. Reenforce edges with direction information, depth information, or region information or relaxation techniques.

    C. Remove noise, thin edges and then thicken uniformly for preparation for depth determination and template matching.

III. Depth map

    A. Match regions using coarse to fine search (See hierarchical search techniques pg. 486 Hall)

        1. Use sequential decision rule (pg. 485 Hall)

        2. Use pairing functions for correlation

    B. Use depth for elimination of false edges

IV. Select suitable target regions for template match

    A. Order edges on their length. edges (See Perkins)

    B. Compile a theta-s plot for the edge.

    C. Correlate the theta-s of the edge under investigation with the taget

part theta-s curve to determine rotation and starting position of template.

V. Match desired template to target region

    A. Find closest target region using depth

    B. Rotate, translate, tilt, interpolate between, and scale Fourier descriptor of template (or pieces of template) and then transform into 2D template.

    C. Match template to region starting at lowest level first

        1. Correlate using pairing functions

        2. Ignore paired point unless neighbors reenforce

VI. Determine closest target part with graspable area

VII. Determine best approach to target part using depth information

VIII. Acquire part

6. APPENDIX B. PROGRAM DESCRIPTIONS

Programs

I. Reduce_image

    A. Capabilities

        1. Low pass filtering (mask selected by passed parameter)

        2. Resampling (parameterized)

        3. Outputs to /tmp/image.level[i+1]

        4. Handles any size image

    B. Assumes

        1. Image in /tmp/image.level[i] (or /tmp/image if initial image)

        2. Image in upper left hand corner (ULHC) of 256 by 256 file

        3. Parameters for size of image mask for filtering, and spacing for resampling specified.

II. Edge_detect

    A. Capabilities

        1. Gives strength and direction

        2. Reasonably fast

        3. Works with texture

        4. Thresholds after edge detection

        5. Outputs to /tmp/image.crude_edge

        6. Handles any size image

    B. Assumes

        1. Image file specified

        2. Image in ULHC

3. Parameters such as image size and threshold given

III. Thin_thick

    A. Capabilities

        1. Eliminates isolated noise points

        2. Thins edges

        3. Thickens edges uniformly

        4. Outputs to image.thick

        5. Handles any size image

    B. Assumes

        1. Input in /tmp/image.edge

        2. Input in ULHC

        3. Size of image passed

IV. Theta-S

    A. Capabilities

        1. Handles any size

        2. Compiles theta-s graph. (See Perkins)

        3. Outputs to /tmp/image.theta-s

    B. Assumes

        1. Image in ULHC

        2. Image consist of binary edges

        3. Input in /tmp/image.edge

V. Depth_map

    A. Capabilities

        1. Determines depth by triangulation comparing search window to search image

        2. Finds depth for edges and neighbor pixels

        3. Helps determine valid edges by comparing depth on both sides of assumed edge

4. Uses Search program

B. Assumes

    1. Image and image.edge available

    2. Image in ULHC

VI. Search

    A. Capabilities

        1. Handles any size window and any size search

            a. Window consists of grey levels and X's (don't cares)

            b. In edge match, search need only be for edges (1's), and adjacent background (0's) and the rest of the window field may be don't cares (FFFF in hex).

        2. Employs correlator with pairing function (counters) which are incremented if pixels match in grey

        3. In binary search counters only incremented if neighboring points also correlate

        4. Handles any size

        5. Uses sequential search strategy (pg 485, Hall)

        6. Creates a map of matched locations

        7. Outputs match map to /tmp/image.match_map

VII. Template

    A. Capabilities

        1. Manipulates (rotates, translates, tilts, interpolates between, scales) 2D templates of desired parts using FD as an intermediate state

        2. Inserts X's (don't cares) around part once transformed into 2D

    B. Assumes

. Library available (/template/part)

2. Part specified

VIII. Template_match

A. Capabilities

1. Uses Template and Search programs to match desired parts to target

B. Assumes

1. Image region in image.edge specified (size and upper left hand coordinate)

2. Thresholds for search specified