Viewer-Centred Representations for Location and Inspection

A. M. Wallace, E. Trucco, M. Diprima, and F. Lavorel

Department of Computing and Electrical Engineering,
Heriot-Watt University, Edinburgh EH14 4AS.

Abstract

We examine the use of viewer-centred representations (VCRs) for object recognition, location and inspection, employed in the framework of advanced inspection systems. We introduce a recognition-oriented VCR of 3D objects and show how to compute it using cluster analysis of approximate visibility spaces. Using the same representational framework, we illustrate the automatic generation of VCRs for feature-based inspection of 3D objects.

1 Introduction

First, we describe the use of cluster analysis to compute VCRs for matching to segmented scene descriptions, and discuss the accuracy and complexity of pose estimation (Section 2). Then we illustrate the automatic generation of VCRs for simulated sensor planning and feature-based inspection of 3D objects (Section 3). The two types of VCR have a common source, a CAD model, and share a description of the visibility space and sensors, and a common software environment. Location and inspection are regarded as complementary parts of an advanced inspection system to steer sensors to optimal positions for tasks such as feature checking or measurement.

Aspect graphs [1] are possibly the most popular form of VCR. However, several objections have been raised against the adequacy of exact, contour-based aspect graphs for machine vision. First, even for moderately complex objects, the graph can become very large and the complexity of 3D matching unacceptable. Second, not all the features included in an aspect need be matched to recognise or define the pose of an object. Third, it is unlikely that all aspects in a contour-based aspect graph correspond to images actually observable by real sensors.

Approximate visibility techniques [2] restrict the set of possible viewpoints to a sphere centered around the object, and build a quasi-regular tessellation or geodesic dome covering the sphere with a discrete grid of viewpoints. Feature visibility from each viewpoint may be computed by

BMVC 1994 doi:10.5244/C.8.33

raytracing a CAD model. Surfaces lead to a definition of aspects as sets of object appearances which share the same combination of visible faces [3]. Such a definition is better suited to match a view to a surface patch description of the kind derived commonly from range data.

2 Characteristic views and 3D location

We define four attributes of views, which, although not involving direct comparisons between objects, can capture salient object properties. A *visibility table* is created by raytracing the CAD model from 20, 80, 320 or 1280 viewpoints. From the original model and the visibility table, four continuous measures are derived for each primitive and each viewpoint.

$$\textbf{Visibility}: v = \frac{A_{vp}}{A_p} \qquad \textbf{Area Ratio}: r = \frac{A_p}{A_o}$$
$$\textbf{Covisibility}: c = \frac{N(C_{vp})}{N(C_p)} \qquad \textbf{Saliency}: s = 1 - \frac{N_{vp}}{V}$$

where A_p is the area of a surface primitive, A_{vp} is the area of a primitive visible in a given view, A_o is the total surface area of the object, $N(C_p)$ is the number of adjacent surface pairs in the model, $N(C_{vp})$ is the number of adjacent primitives visible in a given view, V is the total number of ray-traced views and N_{vp} is the number of views in which a primitive is visible.

Hence, to define the VCR, views are characterised by the partial or complete visibility of primitives, the relative size of primitives, and the proportion of the viewsphere in which the primitive is visible. Covisibility acknowledges the significance of primitive groups (or in this restricted case pairs) in object recognition or location. Previously, we have employed entropy measures for object invocation [4]. A similar approach to define VCRs would be more rigorous, yet not necessarily more effective. Here we have adopted heuristics which represent not only the exact probability of occurrence of features in views, but also measures of how useful these are for object location. For example, pairs of features are necessary to define pose, and segmentation parameters are more accurate in larger patches.

2.1 Deriving a VCR using cluster analysis

Given m viewpoints and n features, S is a rectangular $(m \times 4n)$ similarity matrix whose generic i-th line is

$$v_{i1} \quad r_{i1} \quad c_{i1} \quad s_{i1} \cdots v_{in} \quad r_{in} \quad c_{in} \quad s_{in}$$

where v, r, c and s are the four attributes of the i-th viewpoint defined above. Agglomerative hierarchial clustering is represented in the form of a *dendogram* which shows the progressive agglomeration of views as the correlation coefficient increases. As an example, Figure 1 shows the dendogram obtained from 80 views of the optical stand shown in Figure 2.

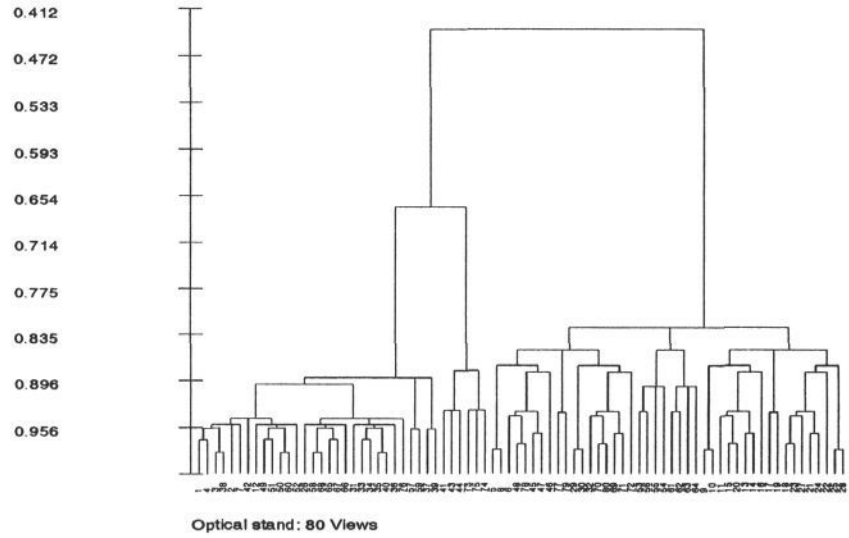


Figure 1: CV dendogram for the optical stand

Five Characteristic Views (CVs) corresponding to a threshold level of 0.840 are shown in Figure 2 (views 18, 47, 53, 73, and 50). Broadly speaking, these 5 views are sufficiently different to avoid unnecessary duplication of the matching process, yet show sufficient simultaneously visible combinations of primitives to allow location of the object in any arbitrary view.

2.2 Using CVs for pose definition

Assuming object recognition or location is based on primitive matching, several constraints may be employed to limit combinatorial complexity, including type definition, unary properties such as surface area, pairwise geometric constraints or concentration on local primitive groups. Here, we examine how feature grouping by CVs affects accuracy or combinatorial complexity of matching and pose estimation.

Each scene or model primitive is represented by a point vector, so that a minimum of two matching primitives is required to solve for the pose. Normal and axis vectors, and central and central axis points are used for planes and cylinders respectively. Pose determination is based on minimisation of the least square error between the model and scene point vectors

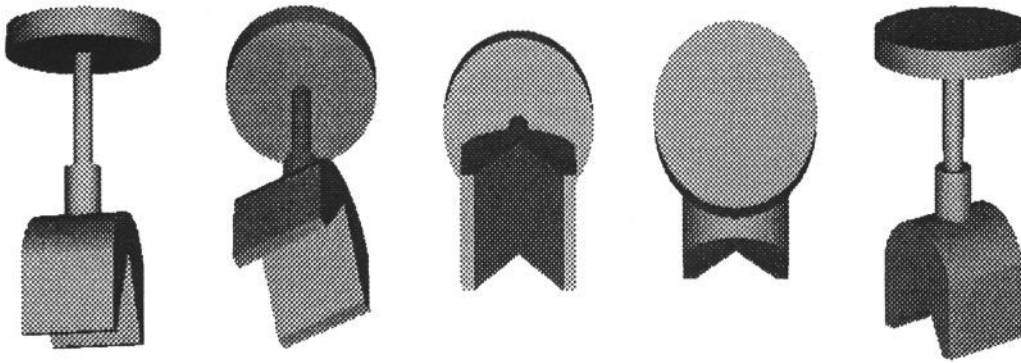
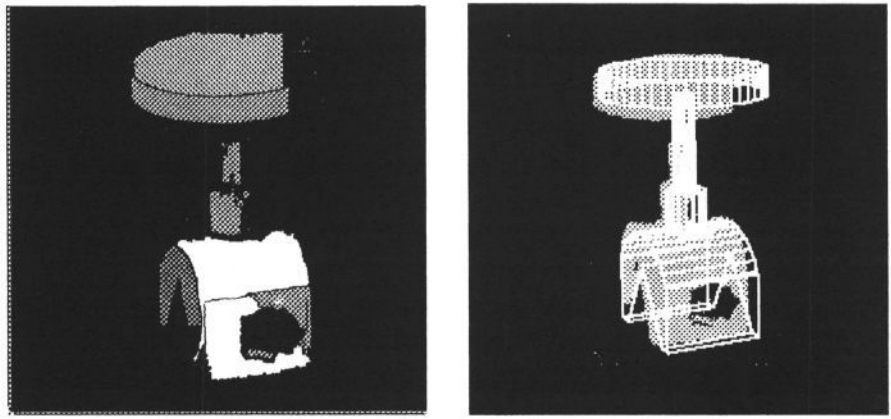


Figure 2: The five CVs of the optical stand generated automatically

using singular value decomposition. For example, Figure 3 illustrates pose definition on the basis of 5 matched primitives between a segmented depth image and a CV. This shows some error in the computed pose, due to the inaccurate definition of the scene control points.



Segmented surface patches Computed pose

Figure 3: Matching the view model to segmented depth data

Comparative accuracy of pose determination: we have compared pose estimates by matching a full object model to random 3D rotations of that model subject to random movement of the feature point vectors. From sets of 1000 trials on a number of modelled objects, we were able to conclude that there was no significant difference between the mean accuracy of the full set of object features and the reduced sets of the CVs, although the standard deviation was greater in the latter case.

The complexity of viewpoint modelling: considering an object with n

features, we can define an upper limit of $\frac{n!}{(n-m)!}$ ways of choosing m model features to match m features in the scene data, as order is important. The probability of finding a correct solution within p random selections, assuming further that the same selection may be chosen more than once, is $1-(1-\frac{n!}{(n-m)!})^p$, which implies that it would take 145 selections of 2 features from a 15 feature model to obtain a 50 per cent chance of the correct selection. Clearly, this is not a sensible strategy, but it does provide a baseline for comparison.

If we assume a known CV model which corresponds to a given view, then this reduces the number of features which may be chosen. For example, if n=8 and m=2, then this leads to a 50 per cent chance of the solution after 38 selections. Thus, if the correct CV is known or may be indexed by a view-specific property of the scene, a reduction in matching complexity is expected.

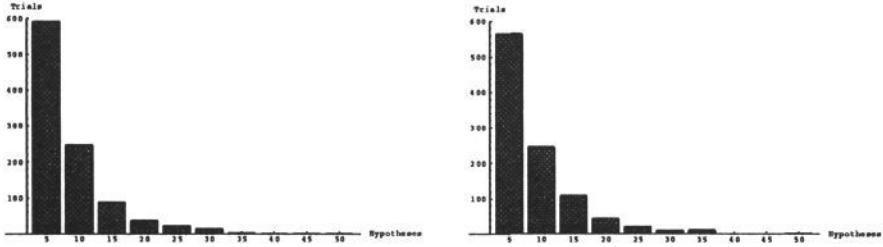
If the correct CV is not known, the situation is more complicated, since all CV's may have some features in common with an arbitrary view, but the appropriate CV is most similar. Assuming pessimistically that the correct pair only occurs in one CV, then we require 194 selections, but this is an unlikely, worst-case model. In general, we might anticipate no advantage over a full object model.

Figure 4 shows histograms from matching an object model and a randomly selected CV of the stand against 1000 random scenes. The horizontal axis depicts the number of hypotheses generated before successful pose definition. The vertical axis shows the number of trials. As anticipated there is no significant difference between the two histograms.

We have also generated similar data for pre-selected view models, anticipating a reduction in complexity. In fact this did not occur as the histograms were substantially the same as those in Figure 4. To explain this, consider the stand illustrated in Figure 2 which is a typical exemplar of the objects tested.

The model has 15 features, but the choice is further constrained. Hypotheses may only be generated for matches between features of similar type, i.e. cylinder to cylinder or plane to plane, and similar radius of curvature for cylinders. There are 55 plane-plane (pp), 6 cylinder-cylinder (cc), and 44 plane-cylinder (pc) choices, each of which may occur in either order, making 210 in all. Of the 110 pp possibilities, there are 86 with non-parallel vectors, but each plane with a vector other than v_1 through the central axis has a symmetric pair. Hence there are 2 out of 86 possible correct solutions. Of the cc cases, only those using the lower, largest cylinder, have non-parallel vectors, and these are constrained by radius compatibility, so that there are 3 out of 3 possible correct solutions. For pc cases, the order is determined by type compatibility, and 17 possibilities are eliminated by

parallel vectors, leaving 8 out of 27 correct solutions, allowing for symmetry. Hence, the probability of selecting 2 point vectors which allow a pose solution at random is 13 out of 116, or 0.112. Thus, a correct match is expected within the first 5 hypotheses in 55 per cent of the cases. This analysis corresponds closely to the data of the leftmost Figure 4 in which about 59 percent are located in the first five hypotheses; the mean number of hypotheses is 6.22.



Object model CVM

Figure 4: Complexity of the matching process

A similar result is obtained in the rightmost histogram. Although there is a considerable variation in the number of visible features, e.g from four (2 planes, 2 cylinders) to ten (6 planes, 4 cylinders) in Figure 2, the mean number of hypotheses is 6.64.

Finally, we consider the case where model invocation has occurred, i.e. there is some characteristic within the view which allows pre-selection of the correct CV. A reduction in complexity was expected, but does not occur. (The mean number of hypotheses is 6.67). In fact, the possible gain of feature grouping by viewpoint is eliminated by two factors. First the constraints imposed by type and value checking are dominant in reducing the complexity of the other cases, and second the object is symmetric, so that it is (almost) invariably possible to find a pair of matching features between a random view and a CV model.

3 Computing and using inspection-oriented VCRs

Once the position of a 3D object in space has been estimated, an advanced inspection system should bring robot-mounted sensors to positions from which a given inspection task can be performed optimally. In general, several parts of an object require inspection, and each is best observed from a different viewpoint. Therefore, a sequence of sensor positioning actions (an *inspection script*) must be planned to perform the task. GASP

(General Automatic Sensor Positioning), a sensor planning system capable of generating inspection scripts for a variety of tasks, objects and sensors, was introduced in [5]. Here we show some recently added modules for simulated optimal sensor positioning.

3.1 An overview of GASP

The FIR (Feature Inspection Representation), makes inspection-oriented information explicit and easily accessible. FIRs are used by GASP to create inspection scripts in a simulated environment [5]. Using the same raytracing process as above, viewpoints are weighted by two sensor-dependent merit coefficients, *visibility* and *reliability*. The former indicates the size of the feature in the image, the latter the expected confidence with which the feature can be detected or measured, based on experimental analysis of image acquisition and analysis. An *optimality* coefficient combines these into a single measure. FIR sizes vary with the number of object features, the resolution of the raytracer images, the density of the viewpoints on the dome, and the regions of space reachable by the sensors.

3.2 Generating inspection scripts

GASP can compute inspection scripts for both single or multiple sensors. A single sensor can be either an intensity or a range imaging camera; the only multiple sensor is a stereo head, a pair of intensity cameras. GASP can generate the following scripts.
(1) *Single-feature, single-sensor:* find the position in space from which a single imaging sensor can inspect a single object feature optimally.
(2) *Single-feature, multiple-sensor:* find the position in space from which a stereo head can inspect a single object feature optimally.
(3) *Multiple-feature, single-sensor:* (3a) find the position in space from which a single imaging sensor can simultaneously inspect a set of features optimally; (3b) find the best 3D path for a single sensor to inspect a set of object features.
(4) *Multiple-feature, multiple-sensor :* find the best 3D path for a stereo head to inspect a set of object features.

Examples of (1) and (3a) were described in [5], so we limit ourselves to (2), (3b) and (4).

3.3 Stereo inspection scripts

For a stereo head, a quantity must be defined expressing how well *both* cameras are placed, since in general good visibility of a feature for one camera does not imply the same for the other. To attach a single optimality

coefficient for each viewpoint, we define the head position as that of the midpoint of the line connecting the optical centres of the cameras. The optimality is the product of the single-sensor optimalities of the two camera viewpoints.

The stereo visibility algorithm takes as input a portion of the property sphere describing a visibility region (of a feature or set thereof), in which every viewpoint is weighted by an optimality coefficient (notice that such a region may *not* be maximally connected). The output of the algorithm is an ordered list of head positions $(c_1, \ldots, c_k)$, $c_i = (V_{il}, V_{ir})$, $V_{il} \neq V_{ir}$, where V_{il}, V_{ir} are the viewpoints of the left and right cameras. The global complexity of the stereo positioning algorithm is $R(O(F) + O(RC)) = O(RF) + O(R^2C)$, where R is the number of viewpoints in the visibility region, C is the number of viewpoints at a fixed distance from a given one (a function of the dome's resolution), and F is the number of viewpoints on the geodesic dome. If $O(F) < O(RC)$, the complexity becomes $O(R^2C)$. Figure 5 shows (shaded) the optimum position for a stereo head of baseline 400mm to inspect the planar face on the bottom side of the object, nearest the viewer, having the inverted 'V' bounding contour. The focal length of the cameras was 20mm, the dome radius (determined by GASP) 1376mm. The stand is about 16cm tall. This solution was found in about two seconds.

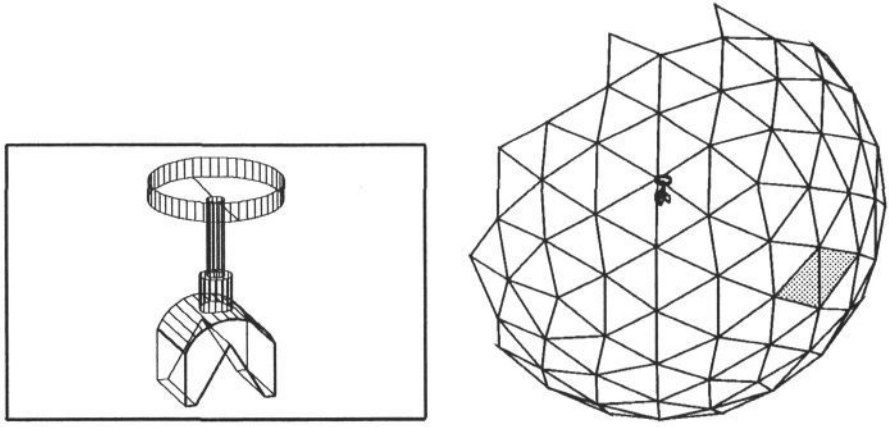


Figure 5: Example of stereo head positioning

3.4 Sequential inspection scripts

From the FIR, we can associate with any set of features F a set of optimal viewpoints $V_F \subset R^3$. One might want to inspect the features in F *sequentially*, for instance because no viewpoint grants simultaneous visibility of all

the features in the set, or because it is desired to ensure maximum visibility or reliability while inspecting each feature. The problem is then to find the shortest path in space to visit all viewpoints in V_F - the travelling salesperson problem (TSP) in 3D. We implemented three existing TSP algorithms, simulated annealing, the elastic net method and the CCAO algorithm [6], We evaluated their performance over a large number of distributions of viewpoints, recording path lengths and distances from the overall optimal solution found (or from the real shortest path when this was known). With up to 100 sites, CCAO outperformed the other two, and was therefore incorporated in GASP. In brief, the CCAO algorithm ("convex hull, cheapest insertion, angle selection and Or-optimisation") combines the features of tour construction and tour improvement TSP algorithms to build an initial path through a subset of V_F, and extends the path incrementally until all viewpoints are visited. The path is then optimised by analysing possible swaps between arcs leading to lower-cost paths.

Figure 6 shows an example of a simple shortest inspection path (complex 3D paths are difficult to visualise) through the optimal stereo-inspection viewpoints of five faces of the stand, namely the four side planes of the base and the top plane of the "seat". Solid lines connect points on visible dome facets, dotted lines involve at least one occluded (not shown) viewpoint. The stereo head is represented by its baseline and the directions of the cameras' z axes. This solution was found in less than a second.

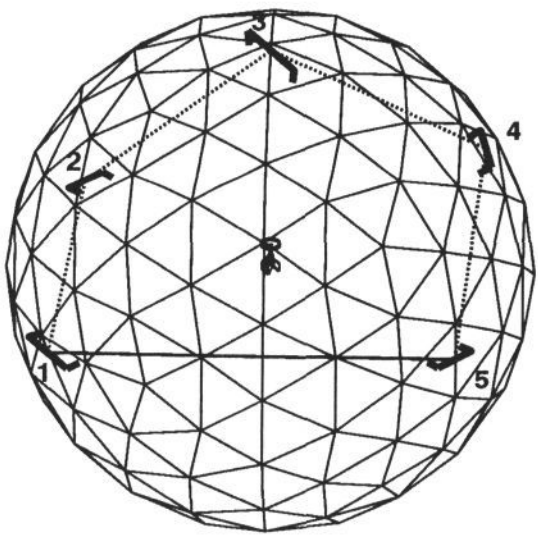


Figure 6: Example of inspection path for a stereo sensor

4 Conclusions

Although the CVs within the VCR provide a sufficient subset of views for recognition and location of arbitrary views of rigid objects, the use of "view groups" or subsets of features is not generally effective in reducing complexity in primitive matching schemes. Other constraints, applicable alike to object or view models, such as type checking and pairwise geometry provide better results. This must be qualified; view-dependent invariants may still have a role to play in object recognition for a large set of possible models, but such invariants must be robust and ensure rapid generation of valid hypotheses.

Viewpoint modelling appears well-suited to support a variety of applications in visibility based sensor planning, including inspection of single and multiple surface features using both single and stereo sensors. Combining recognition, location, planned inspection, and fuller sensor modelling within a single framework is a useful ability of an advanced visual inspection system.

References

[1] K.W. Bowyer and C.R. Dyer. Aspect graphs: an introduction and survey of recent results. *International Journal of Imaging Systems and Technology*, 2:315–328, 1990.

[2] G. Fekete and L.S. Davis. Property spheres: a new representation for 3-d object recognition. In *Proc. IEEE Workshop on Computer Vision, Representation and Control*, pages 192–201, 1984.

[3] K. Ikeuchi and K.S. Hong. Determining linear shape change; towards automatic generation of object recognition programs. *Computer Vision, Graphics and Image Processing*, 53(2):154–170, 1991.

[4] A. M. Wallace and P. McAndrew. Inferring the presence of objects from feature data. *Pattern Recognition Letters*, 9:287–295, 1989.

[5] E. Trucco, E. Thirion, M. Umasuthan, and A.M. Wallace. Visibility scripts for active feature based inspection. In *Proc. of British Machine Vision Conference*, pages 538–547, 1992.

[6] B.L. Golden and R. Stewart. Empirical analysis and heuristics. In *in The Travelling Salesman Problem, ed. Lawler, Lenstra, Rinnoykan, Shmoys.* Wiley, 1985.