

OCCLUSION ANALYSIS USING BOUNDARY CORRESPONDENCE

T. Rachidi* and L. Spacek

Department of Computer Science
University of Essex
Colchester CO4 3SQ
U.K.

Abstract

The occlusion problem has long been considered at the 3-D level of scene analysis. The present work however, brings a new insight into it, and shows that it can be considered at the 2-D level without the help of predefined models. Provided that a correspondence is established between boundaries in successive frames, and the appropriate assumptions are made, boundary pairings can be used to give a $2\frac{1}{2}$ -D interpretation to the scene being viewed. Occlusion junctions are at the heart of this interpretation. A scheme for analysing local displacements at these junction points is presented, together with a propagation technique which solves occlusion and clusters boundaries of the same physical object. This analysis shows that motion information is not always needed to solve occlusion.

1 Introduction

Occlusion constitutes a fundamental part of motion analysis since every realistic scene contains occlusion. The pattern of occlusion itself depends upon the *view-point* and the relative positions of the objects in the scene. The human visual system excels at coping with occlusion and inferring the relative position/depth of objects from it. Successful Machine Vision systems are expected to cope with occlusion in a knowledge-free fashion. Such systems are expected to generate answers like “the set of boundaries/regions¹ S_1 is a single object and is occluded by the object given by the set of boundaries/regions S_2 ”. This type of scene description is totally general, and can be used by higher level processes in order to establish a human-like perception of the scene.

*The author is now with the Faculty of Science and Engineering, Al-Akhawayn University in Ifrane, Hassan II Avenue, PO. BOX 104, 53 000 Ifrane, Morocco.

¹By definition, boundaries and regions are dual, *i.e.*, one description/representation leads to the other. However, this duality may not be total in places where boundaries connectivity is broken.

This description of the scene cannot be achieved without occlusion analysis/detection. The latter has always been considered at the 3-D level of scene analysis. Often, occlusion analysis is thought of as a task which can only take place when a full 3-D model of the scene has been recovered. This may historically stem from the first attempts to give 3-D interpretations to line drawings in single images [9, 5, 7, 17, 11, 8, 6].

Furthermore, occlusion analysis has been associated with predefined 2-D models (polygonal figures) and 3-D models (blocks world). Line segments in the image are extracted together with their connectivity, and matched against stored models allowing for partial correspondence. Unmatched lines of the model are marked as occluded.

In this paper, we show that the establishment of correspondence [12] between boundaries in two successive frames can yield quality information about the scene in general, and occlusion detection in particular.

2 Related work

Aggarwal & Duda’s system uses motion information to analyse occlusions of rigid polygonal figures that are moving with various 2-D velocities [1]. The system requires angular movements between images, and two additional constraints:

- an acute angled vertex cannot be generated by two polygons, *i.e.*, it must belong to some actual polygon;
- at most one vertex may become occluded or become visible between any two frames.

Due to the rigidity assumption, non-occluding edges of each constituent object should exhibit similar motions. Thus, by associating edge segments with corresponding movements, the apparent object can be analysed into its constituent objects.

In their algorithm for interpreting sequences of line drawings, Asada *et al.* limit themselves to the blocks world [2]. One underlying assumption is the use of the

orthogonal projection, *i.e.*, the 2-D changes obtained from images convey the nature of movements in the 3-D scene.

A major drawback of using line drawings is the limitation to the polyhedra world, and the development of techniques which are not robust or versatile enough to cope with natural scenes [4]. Natural scenes are so rich in details that their representation with line drawing would be very complex. The crucial factor is how to select those boundaries representing significant details for the process in hand. This problem has no solutions as yet. Even after simplifying images to a subset of perfect line-drawings, additional information about the world is needed for inferring the physical nature of boundaries.

Many researchers have worked with imperfect input when interpreting/labelling boundaries in a single image [7, 15, 16, 10]. Simple cases were considered, such as where lines and junctions are missing, and parallel lines are not well drawn. No attempt was made to consider real input, in which imperfections are rather the norm than the exception. Cooper extends the classical Waltz labelling scheme to include C-junctions [17, 6]. However, the list of assumptions for the approach is long and unlikely to be satisfied by real input.

In general, occlusion analysis has been limited to known objects in the scene, and/or has been performed as a 3-D stage on static image(s). In this work, we intend to show that occlusion analysis can be performed at the 2-D stage, without *a priori* knowledge of the scene, by means of boundary correspondences. We assume that topology is conserved *almost everywhere* between two subsequent frames. The topology constraint is explicitly embedded in the boundary-based correspondence computation in order to yield correct matches between boundaries [12].

To this end, junction points are locally analysed, and the local information is propagated consistently across neighbouring boundaries. There are some cases where no information can be derived². These cases are identified and the corresponding junctions are assumed to be solid. The general aim of our approach is to recover as much information as possible, in a data-driven fashion.

3 Junction point analysis

Although, the illustrations and discussion in this section are presented for 3-way junctions, the reasoning is the same for all types of junctions. The apparent limitation to 3-way junctions in the present work is for sake of clarity. However, note that the other types of junctions

²For instance, if there is no motion at a junction point, boundaries at this junction can equally be part of two different physical objects, or part of a single object (a cube corner for instance). This ambiguity also arises when the relative motion of the occluded and occluding objects is nil. In most of these situations, the human visual system also fails to perceive what is actually happening whilst using motion clues alone.

are extremely rare in motion sequences, and arise from accidental alignments.

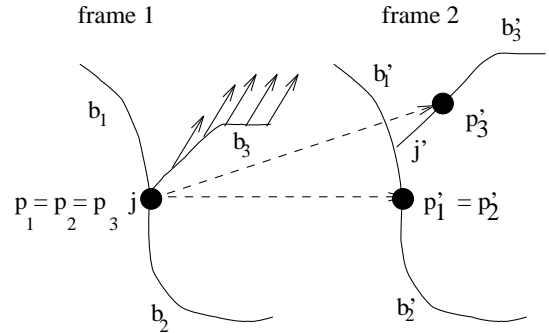


Figure 1. *Analysing junctions of occluding boundaries: The junction point j depicts 3 physical points p_1, p_2 and p_3 ; $p_1 \in b_1, p_2 \in b_2$, and $p_3 \in b_3$. Solid arrows represent motion against the image. Dashed arrows indicate the movements of each of the three physical points. $p'_1 = p'_2$, since (b_1, b_2) are part of the same physical object.*

We observe that, at occlusion locations, junction points (such as j in **Figure 1**) are seen to ‘slide’ ($j \rightarrow j'$) along the two occluding boundaries (b_1, b_2). Identifying these two boundaries and marking them as part of the same physical object is the heart of the occlusion problem, and is the task we intend to solve.

An occlusion junction point j is actually an abstract point which can be decomposed into three physical ones: $p_1 \in b_1, p_2 \in b_2$ and $p_3 \in b_3$. By identifying these physical points on the set of boundaries $\{b'_1, b'_2, b'_3\}$ of the subsequent frame, one also identifies the two boundaries out of the three, that are physically joined as parts of the same occluding object.

Identifying points p'_1, p'_2 and p'_3 in the second frame is equivalent to identifying portions of boundaries $j'p'_1, j'p'_2$ and $j'p'_3$ which appear or disappear between frames. One way of doing this is by finding the longest sub-matches between the pairs $(b_1, b'_1), (b_2, b'_2)$ and (b_3, b'_3) . However, this is an expensive process, and requires at least the *a priori* knowledge of the transformation each boundary undergoes.

Suppose that only the occluded object moves, as in **Figure 2**. By superimposing point j on *frame 2*, one can easily check to which of the boundaries $\{b'_1, b'_2, b'_3\}$, j belongs. Similarly, by superimposing point j' on *frame 1*, one can determine to which of the boundaries $\{b_1, b_2, b_3\}$, j' belongs (see **Figure 2**).

Formally, the following analysis is performed for every set of three boundaries meeting at a junction:

Let B be the set of boundaries of *frame 1* having a match in *frame 2*. $B = \{b_i\}_{i=1..n}$.

A subset $E \subset B$ is called a *consistent set over frame 1* if and only if $E = \{b_1, b_2, b_3\}$ and b_1, b_2 and b_3 meet at a junction point j_E and $cr(b_i) = b'_i \forall i = 1..3$, where $cr(e)$ denotes the element which matches e in *frame 2*.

Let $E' = \{b'_i \in B' \mid b'_i = cr(b_i) \forall b_i \in E\}$. Because the matches are topologically correct, E' is a *consistent set* over *frame 2*. Let also $j_{E'}$ be the junction point where b'_i s meet (see **Figure 2**).

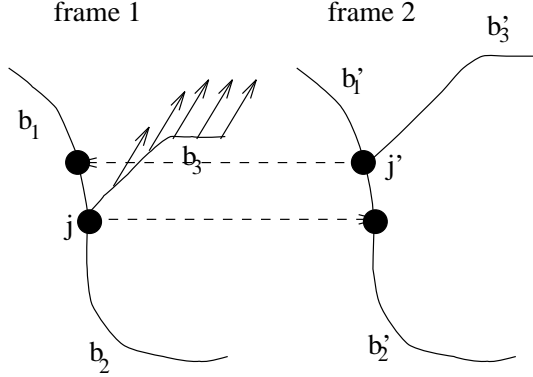


Figure 2. $E = \{b_1, b_2, b_3\}$ and $E' = \{b'_1, b'_2, b'_3\}$ are examples of consistent sets. (b_1, b_2) are part of the same physical object.

Obviously, if $j_E = j_{E'}$, either no movement has occurred at the location under focus, or a movement has occurred but did not result in any perceptible changes in the location under focus (rotation, for instance).

Let E and E' be two corresponding *consistent sets* over *frame 1* and *frame 2* such that $j_E \neq j_{E'}$.

We define the function $belongs(j_{E'}, E)$ as follows:

$$belongs(j_{E'}, E) = \begin{cases} b & \text{if } \exists b \in E \mid \text{the point } j_{E'} \in b \\ \emptyset & \text{otherwise} \end{cases}$$

This function simply returns the boundary in the *consistent set* E which the junction point $j_{E'}$ belongs³ to. If this junction point does not belong to any boundary in E , the function returns the empty set. In the example of **Figure 2**, $b_1 = belongs(j', E)$ and $b'_2 = belongs(j, E')$.

Let $b_k = belongs(j_{E'}, E)$ and $b'_h = belongs(j_E, E')$. Four scenarios can be identified:

³This is done by finding the nearest point A on the boundary to the point $j_{E'}$. $j_{E'}$ is declared belonging to the boundary if $d(A, j_{E'}) \leq 2$ pixels, to take account of noise problems.

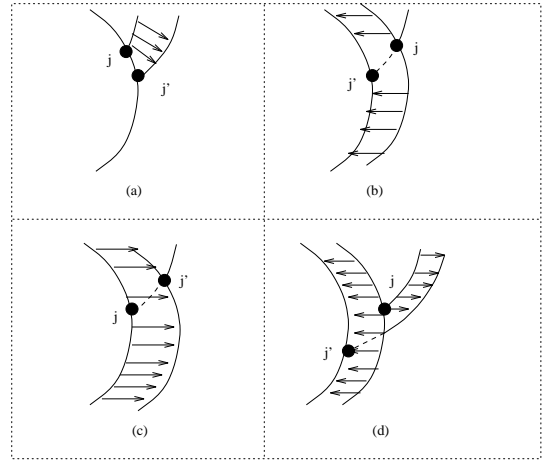


Figure 3. *The four scenarios. Solid arrows indicate motion against the image. Dashed arrows indicate the location of j_E in E' , and $j'_{E'}$ in E . Boundaries b_k and b'_h are respectively in frames 1 and 2.*

- $b_k \neq \emptyset$ and $b'_h \neq \emptyset$:
this scenario happens when only the occluded object has moved. A *sliding* occurs along the contour $\{b_k, cr^{-1}(b'_h)\}$. Hence, the two boundaries $\{b_k, cr^{-1}(b'_h)\}$ are part of the same physical object, and are to be marked so. Let $R = \{b_k, cr^{-1}(b'_h)\}$. (see **Figure 3** (a))
- $b_k = \emptyset$ and $b'_h \neq \emptyset$:
this scenario occurs when the occluding object alone moves to uncover the occluded object. Let $R = E \ominus \{cr^{-1}(b'_h)\}$. R is a set of two boundaries belonging to the same physical object, and are thus to be marked as such. (see **Figure 3** (b))
- $b_k \neq \emptyset$ and $b'_h = \emptyset$:
this scenario happens when the occluding object alone moves to further eclipse the occluded one. Let $R = E \ominus \{b_k\}$. R is a set of two boundaries belonging to the same physical object, and thus are to be marked as such. (see **Figure 3** (c))
- $b_k = \emptyset$ and $b'_h = \emptyset$:
this occurs when both the objects are moving with respect to the image (see **Figure 3** (d)). It may indicate that the three boundaries are part of the same physical object, *i.e.*, the junctions j_E and $j'_{E'}$ are not occluding junctions. It can also happen in the context of independently moving objects. To decide which of the boundaries are part of the same physical object, motion information is used. Boundaries of the same physical object have the same *3-D* motion parameters. However, computing the *3-D* motion of each boundary requires the depth information which is not available. Since the purpose of this research is to demonstrate that boundary correspondence can yield an accurate occlusion analysis (and hence object separation), we assume that the *3-D* motion of boundaries is captured by the *2-D* model of [13]. Note that when the motion of boundaries cannot be expressed

by the 2-D model, so rather than performing a hazardous marking, no marking is performed at that junction in accordance with the least commitment principle.

Let $\mathfrak{R}_{b_i} = k_{b_i} \cdot R_{b_i} \circ T_{b_i}$ be the 2-D motion of boundary $b_i \in E$, $i = 1, 2, 3$. The computation of the components k , R and T of \mathfrak{R} is given in [13]. Let $P_i = \mathfrak{R}_{b_i}(j_E)$, $i = 1, 2, 3$ be the points obtained by applying each boundary motion to the junction point j_E (see **Figure 4**).

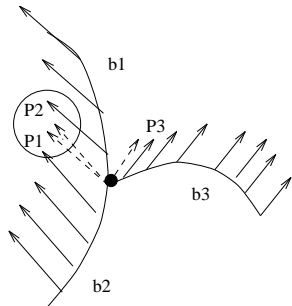


Figure 4. *Decision strategy for marking boundaries of the same object. The motions of boundaries are shown in solid arrows. Dashed arrows indicate the points P_1, P_2 and P_3 . b_1 and b_2 are marked part of the same physical object.*

The two boundaries which are part of the same physical object have their corresponding P_i falling in the same neighbourhood. If, however, all the three points $P_i, i = 1, 2, 3$ fall in the same neighbourhood, the junction is taken to be a ‘solid’ one, and the three boundaries are marked part of the same physical object. This situation may also arise when two overlapping objects are undergoing the same motion. The human visual system is also incapable of separating occluded objects undergoing the same motion.

Note that scenario (d) does not occur in the context of moving objects against a static background, and that motion can also be used in the other three scenarios. Yet, to keep the system simple and minimise errors, the use of this feature is limited to a special case in the present implementation (see **Figure 3** (d)). The simple superimposition technique is used to identify points p'_1, p'_2 and p'_3 in all the three cases.

4 Taking into account all junctions: Propagation

The marking of a boundary at one junction has to be consistent with the other marking at the neighbouring junctions (see **Figure 5**). This dictates a *boundary colouring scheme* (see *Blob Colouring* in [3]) to propagate the colour of connected boundaries of the same physical object across junctions.

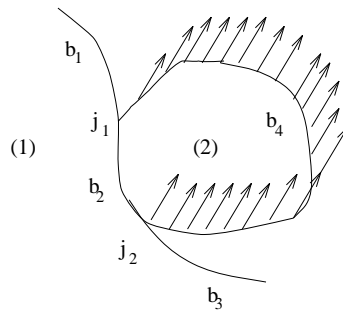


Figure 5. *By analysing j_1 , (b_1, b_2) are to be marked as part of the same physical object. By analysing j_2 , (b_2, b_3) are to be marked as part of the same physical object. The colour of b_2 from j_1 is to be consistent with the one from j_2 . Solid arrows indicate motion of object (2) against the image.*

Let $Colour(e)$ be a colour associated with each boundary $e \in B$ such that:

$$Colour(e) = \begin{cases} -1 & \text{if the colour of } e \text{ is unset} \\ id & \text{if the colour of } e \text{ is set to } id \end{cases}$$

Initially, all boundaries have their colours unset, *i.e.*, $\forall e \in B Colour(e) = -1$.

Let $R = \{b_i, b_j\}$ be a set of two boundaries to be marked belonging to the same physical object. Four situations can occur:

- $Colour(b_i) \neq -1$, and $Colour(b_j) = -1$: propagate b_i 's colour by $Colour(b_j) \leftarrow Colour(b_i)$.
- $Colour(b_i) = -1$, and $Colour(b_j) \neq -1$: propagate b_j 's colour by $Colour(b_i) \leftarrow Colour(b_j)$.
- $Colour(b_i) = -1$, and $Colour(b_j) = -1$: set both colours of b_i and b_j to a new colour id , $Colour(b_i) \leftarrow id$, $Colour(b_j) \leftarrow id$.
- $Colour(b_i) \neq -1$, and $Colour(b_j) \neq -1$: $\forall e \in B$ such that $Colour(e) = Colour(b_i)$ or $Colour(e) = Colour(b_j)$ do $Colour(e) \leftarrow id$, where id is a new colour.

Unlike some *consistent labelling schemes* for interpreting/labelling 2-D scenes which require search techniques with backtracking [11, 8], the boundary colouring scheme is simple and straightforward.

5 Identifying object's regions

At the end of the propagation, *i.e.*, boundary colouring, boundaries with the same colour are joined into sets called *clusters* $(G)_i, i = 1..g$. Clusters represent sets of boundaries which belong to the same physical object.

On the other hand, from a topological point of view, an object in the scene is a collection of regions (in Machine Vision terms). Each region is enclosed within a set of boundaries. Let $(R)_i, i = 1..r$, be the set of all regions. Henceforth, a region R_i is assimilated to the set of boundaries enclosing it. Boundaries such as b_2 in **Figure 5** belong to two different regions, by construction of the latter. To decide which region such boundaries belong to, a *subtraction* scheme is adopted, in which the clusters $(G)_i$ are used to *subtract* unwanted boundaries from the regions $(R)_i$.

The idea behind this *subtraction* is that boundaries of the same cluster $(G)_i$ either all belong to a given region or none do.

Subtract $((G)_i, (R)_j)$
 GROUPS G_i ;
 CYCLES R_j ;

```

BEGIN
  FOR  $i \in [1..g]$  DO
    FOR  $j \in [1..r]$  DO
      IF  $R_j \cap G_i \neq G_i$  THEN
         $R_j \leftarrow R_j \ominus (R_j \cap G_i)$ 
    END
  END

```

Algorithm 5.1. *Subtraction: either all boundaries of a cluster belong to a given region or none do.*

After subtraction, the remaining boundaries of each region are part of the same physical object. For instance, suppose that object **(1)** of **Figure 6** undergoes a translation, we have:

- The set of clusters $G_1 = \{B_1, B_4, B_6\}$ emerging from the analysis of junctions j_1 and j_2 , and $G_2 = \{B_2, B_3\}$ resulting from the analysis of junction j_3 and j_4 .
- The set of regions $R_1 = \{B_0, B_1, B_5, B_2\}$, $R_2 = \{B_6, B_1, B_4\}$ and $R_3 = \{B_2, B_3\}$.

The *subtraction* stage leaves the regions $R_1 = \{B_0, B_5\}$, $R_2 = \{B_6, B_1, B_4\}$ and $R_3 = \{B_2, B_3\}$, which are indeed sets of boundaries belonging to the same physical objects.

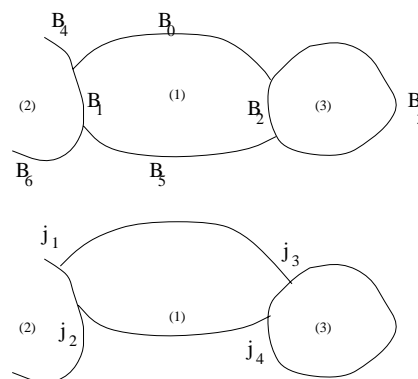


Figure 6. *Example of subtraction. Object (1) undergoes a translation to the top. See text for explanation.*

Figure 8 shows the occlusion analysis obtained for the artificial sequence shown in **Figure 7**.

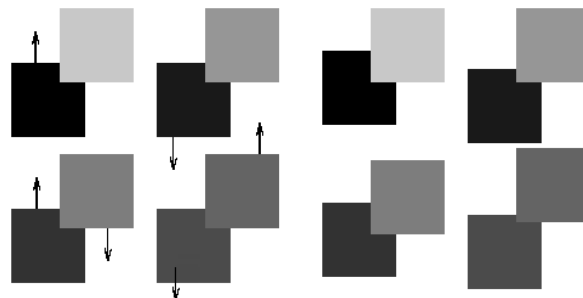


Figure 7. *An artificial sequence: motion between the two frames is indicated by arrows.*

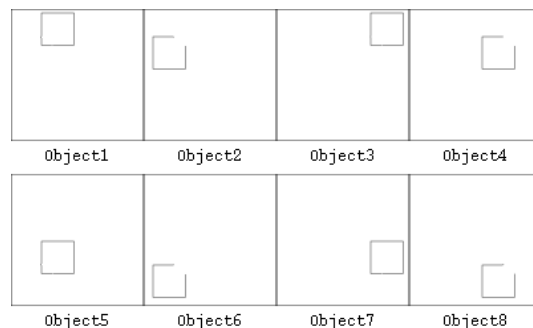


Figure 8. *The results of the occlusion analysis performed on the artificial sequence in Figure 7. Objects detected are separated into different images.*

Figure 10 shows the occlusion analysis obtained for the artificial sequence in **Figure 9**. The aim of these experiments is to test the occlusion based colouring scheme.

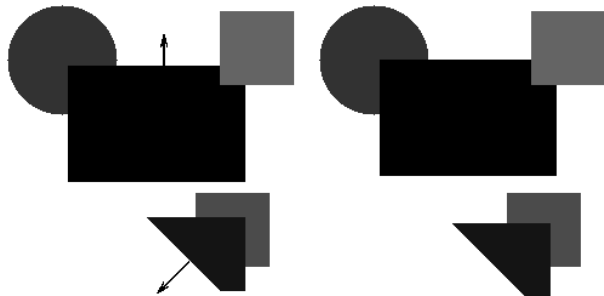


Figure 9. An artificial sequence: motion between the two frames is indicated by an arrow.

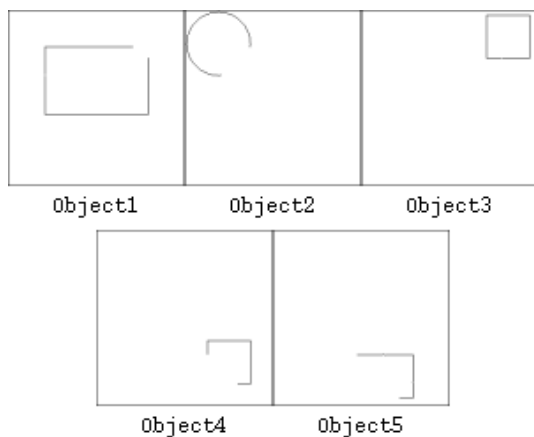


Figure 10. The results of the occlusion analysis performed on the artificial sequence in Figure 9. Objects detected are separated into different images.

However, the *subtraction* scheme is not enough to assemble objects with many different regions. Regions which share a common boundary are fused into a single cluster (see **Algorithm 5.2**). This procedure is repeated until no further clustering is possible. It underlies the following assumption: Regions are assumed to be part of the same object unless there is evidence to the contrary. Consequently, junctions where no displacement is detected are assumed to be “solid” junctions.

```

Cluster((R)i)
REGIONS Ri;

BEGIN
  FOR i ∈ [1..r] DO
    FOR j ∈ [1..r] DO
      IF Ri ∩ Rj ≠ ∅ THEN
        Ri ← (Ri ∪ Rj)
    END
  END

```

Algorithm 5.2. Clustering: regions sharing boundaries are clustered together.

In order to complete the separation, we need to be able to further merge these clusters to allow for

objects separated into different sections without common boundaries as is in the case in **Figure 11**. This is explained in the following section.

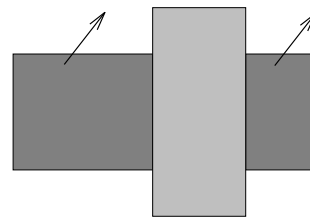


Figure 11. Need for further merging.

6 Object separation in cluttered scenes

Consider now the sets of transitive closures⁴ defined on B (the set of boundaries in image p having matches in image *frame 2*), $(T)_i$ $i \in 1..t$. The *refined* regions of each transitive closure can be further compared for similar motion features for a further clustering.

In the absence of the full $3-D$ motion of boundaries, the $2-D$ motion of a region (*i.e.*, a set of boundaries) cannot be simply established from the motion of its boundaries. Boundaries of a slanting region/surface for instance, have different angles of rotation. Hence, taking the average rotation angles of boundaries as the angle of rotation of the region would be erroneous. The centers of rotation of each boundary of a refined region however, are more likely to be at the same location (which is the projection of the center of rotation of the object the boundaries are part of in the scene onto the image plan). A unique center can be associated with each refined region. Regions with the same center of rotation are most likely part of the same solid object, and can be clustered together.

Additionally, the $2-D$ translation component, when reliably computed can also be used as a feature for clustering regions of the same transitive closure. The translation component of a clustered region can be defined as the translation of its boundaries when the latter are similar in orientation and magnitude. If, however, boundaries of the same clustered region have different orientations and magnitudes, no $2-D$ translational motion is associated with it. This would have the effect of eliminating hazardous clustering.

7 Experimentation results

The aim of this section is to show that the separation of objects in the scene can be performed in a *bottom-up* fashion. Boundaries are first detected using [14]. Then

⁴Given a graph, the transitive closure of a node x is the set of all nodes reached from x .

boundary correspondence is established for two successive frames [12]. Where necessary, the 2-D motion is computed for each boundary in the first frame [13] before feeding the result of the correspondence to the separation module.

Experiment **Figures 12–15** shows the results of the separation, in the case of a fixed camera and independently moving objects.

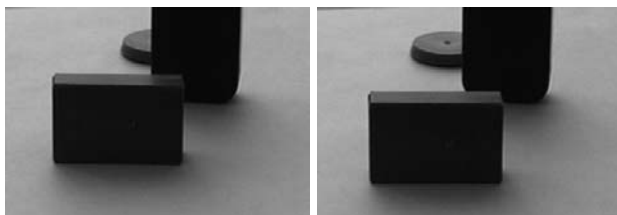


Figure 12. *A real sequence: the object at the back undergoes motion to the left. The object in front moves towards the camera.*

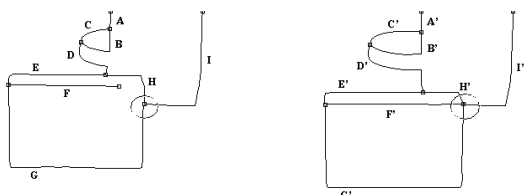


Figure 13. *Boundaries detected for each image of the sequence above. Alphabetic labels are superimposed for discussion in the text. Junction points are marked with squares.*

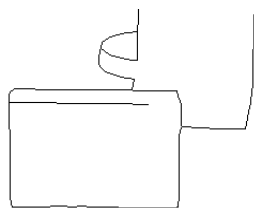


Figure 14. *All boundaries are correctly matched.*

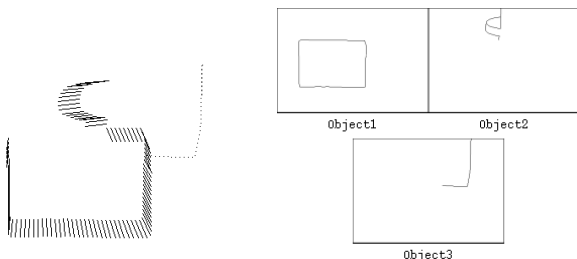


Figure 15. *Left: Motion field obtained for the sequence. Right: the result of the separation. Detected objects are separated into different images.*

Due to an accidental alignment and to a very low contrast, topology has not been totally conserved between the two frames (circled junctions **Figure 13**). Additionally, boundaries between the two objects further from the camera have been over-segmented, due to a very-low contrast (boundaries B, B', D and D'). It should be noted that no special lighting conditions or precautions were taken when grabbing the sequence. Nevertheless, correct correspondence has been established between boundaries (see **Figure 14**).

The object in front moves towards the camera. The motions (see **Figure 15**) of most of its boundaries are well captured by the 2-D affine model [13]. A mere difference of 0.05 is found between expansion factors of boundaries H and G. The parameters found for boundaries C and D are respectively $\theta = 0^\circ$, $k = 1.00$, $X = (-25, 2)^T$ and $\theta = 0^\circ$, $k = 1.00$, $X = (-26, 3)^T$. Note, however, that we could not compute the motion of boundaries F and E. This is due to the fact that such boundaries are “simple”. The motion of the over-segmented boundary, B, also could not be captured by the model. Yet, overall, a precise velocity field for the sequence is computed from boundary correspondence.

The occlusion analysis of the sequence yields the separation displayed in **Figure 15**. Motion information (case (d)) has been used at the junction point (CDB). The rest of the junctions were solved using (cases (a), (b) and (c)). Three objects are correctly identified. Yet, the vertical part of boundary B is identified with the wrong object. Given the sub-optimality in boundary B (B actually represents two different physical boundaries), this is the best possible result that can be achieved. In conclusion, this experiment shows that despite the multitude of sub-optimality due to low level processing, a good separation and motion field can still be achieved.

The following experiment (**Figures 16–19**) shows the results of the separation in the context of the conveyor belt. The conveyor belt was simulated by a mouse pad on which the objects were placed. The mouse pad was then moved manually between two successive frames.

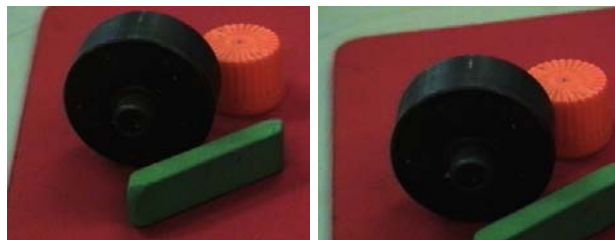


Figure 16. *A simulated conveyor belt sequence: As the mouse pad is manually moved, the different objects undergo a 3-D translation towards the camera.*

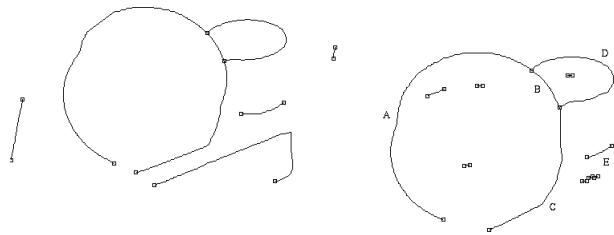


Figure 17. Boundaries detected for each image of the above sequence. Junction and end points are marked with squares.

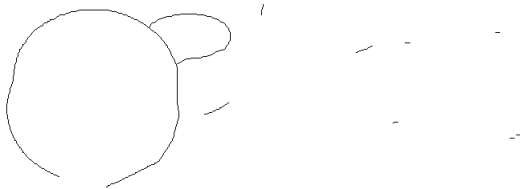


Figure 18. Matched (left) and unmatched (right) boundaries.

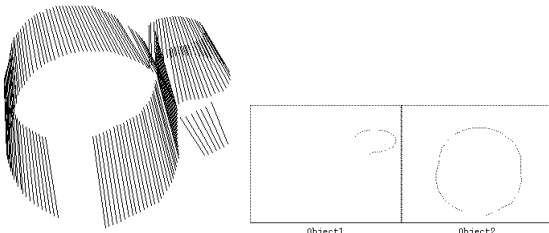


Figure 19. Left: Motion field obtained for the sequence. Right: the result of the separation. Detected objects are separated into different images.

The occlusion analysis of this sequence has led to the separation in **Figure 19**. Both junctions have been solved using motion information. A propagation has taken place, across the two junctions, between the three boundaries of *Object1*. The latter is identified as occluding *Object2*. The occlusion analysis has not been able to detect the third object on the moving mouse pad for an obvious reason: the boundaries of this object have not been properly segmented by earlier stages *i.e.*, the initial edge detection; this is due to the very low contrast between this object and *Object2*.

The experiment in **Figures 20–23** shows a case where the separation was not totally successful. This is because the motion of the solid junction points *J* and *K* occurred along the edges *A* and *B*. This is a case of accidental alignment. The information present at these junction points is not sufficient to determine their nature *i.e.*, solid junctions or intersections between different objects. The separation presented in **Figure 23** underlies an assumption that in such cases we assume the existence of two different objects. Note that motion

parameters could not be computed for the boundaries involved; boundary *A*, for instance is “simple”.

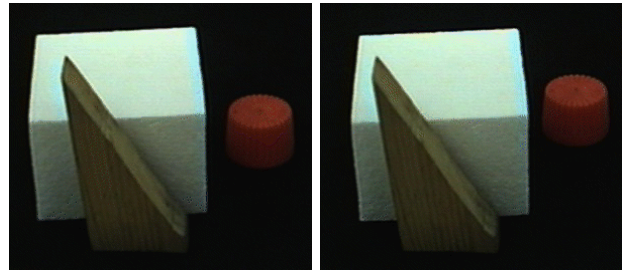


Figure 20. The white box undergoes a translation to the right between the two frames. At the same time, the cylindrical object undergoes a translation away from the camera.

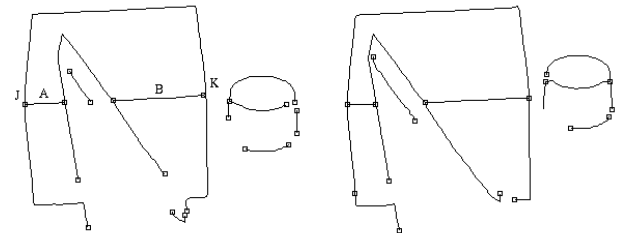


Figure 21. Boundaries detected for each image of the above sequence. Junction and end points are marked with squares.

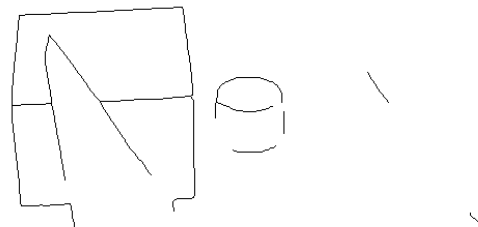


Figure 22. Matched (left) and unmatched (right) boundaries.

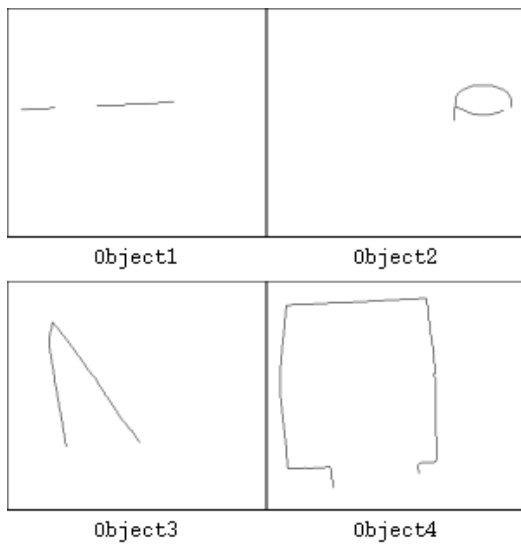


Figure 23. *The result of the separation: Object1 and Object 4 are actually the same object (see text for explanation).*

Conclusion

We have presented a method for the separation of objects in the scene. We showed that analysing occlusion junctions is at the heart of this separation. Unlike some *consistent labelling schemes* for interpreting/labelling *2-D* scenes, which require search techniques with backtracking, our technique is simple, straightforward and works on real images. We particularly showed that the knowledge of the full *3-D* motion of boundaries in the scene is not always needed to fully solve occlusion.

We have also established a partial ordering of objects in the scene, whereby occluding objects (cluster containing an occluding boundary) are nearer to the camera than occluded objects. This ordering is of enormous practicality in robotics applications.

Acknowledgements

This work was partially supported by an ORS award.

References

[1] J. K. Aggarwal and R. O. Duda. Computer analysis of moving polygonal images. *IEEE Transactions on Computers*, c-24:966–976, 1975.

[2] M. Asada, M. Yachida, and Saburo Tsuji. Three dimensional motion interpretation for the sequence of line drawings. In *Proceedings of the 5th International Conference on Pattern Recognition*, pages 1266–1273, 1980.

[3] D. H. Ballard and C. H. Brown. *Computer Vision*. Prentice Hall International, 1982.

[4] Harry G. Barrow and Jay M. Tenenbaum. Retrospective on interpreting line drawings as three dimensional surfaces. *Artificial Intelligence*, 59:71–80, 1993.

[5] M. B. Clowes. On seeing things. *Artificial Intelligence*, 2(1):79–116, 1971.

[6] Martin C Cooper. Interpretation of line drawings of complex objects. *Image and Vision Computing*, 11(2):82–90, 1993.

[7] G. Falk. Interpretation of imperfect line data as three-dimensional scenes. *Artificial Intelligence*, 3:101–144, 1972.

[8] R. M. Haralick and L. G. Shapiro. The consistent labelling problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):173–184, 1979.

[9] D. A. Huffman. *Machine Intelligence 6*, chapter Impossible Objects as Nonsense Sentences. Edinburgh University Press, 1971. Ed., B. Meltzer and D. Michie.

[10] K. Kanatani. Reconstruction of consistent shape from inconsistent data: optimization of $2\frac{1}{2}$ d sketches. *International Journal of Computer Vision*, 3(4):261–292, 1989.

[11] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[12] T. Rachidi and L. Spacek. Boundary-based correspondence computation using the topology constraint. In *Proceedings of the 5th British Machine Vision Conference*, volume 1, pages 55–64, York, September 1994.

[13] T. Rachidi and L. Spacek. Computing 2-d affine transform from points of significant curvature. CSM 224, Dept of Computer Science, University of Essex, Colchester C04 3SQ UK., 1994.

[14] T. Rachidi and L. Spacek. Constructing coherent boundaries. In *Proceedings of the 5th British Machine Vision Conference*, volume 1, pages 296–304, York, September 1994.

[15] R. Shapira. More about polyhedra-interpretation through constructions in the image plane. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(1):1–16, 1985.

[16] K. Sugihara. *Machine Interpretation of line drawing*. MIT Press, Cambridge MA, 1986.

[17] D. Waltz. Understanding line drawings of scenes with shadows. In Winston P. H., editor, *Psychology of Computer Vision*, pages 19–91, New York, 1975. McGraw-Hill.