Moving Object Graphs and Layer Extraction from Image Sequences

David Tweed and Andrew Calway
Department of Computer Science
University of Bristol, UK
{tweed,andrew}@cs.bris.ac.uk

Abstract

We describe a new approach to extracting layered representations from image sequences based on *moving object graphs* (MOGs). A MOG is a form of region adjacency graph which links together local motion segmentations corresponding to distinct moving regions in the scene, typically either foreground objects or the background. The local motion segmentations are obtained by fusing colour segmentations with block motion estimates and the MOGs link segmentations with consistent spatial and motion properties. Linking MOGs across frames then allows temporal consistency to be imposed and layers to be extracted. The approach provides a flexible framework within which to combine local and global constraints both spatially and temporally, enabling robust motion segmentation and layer extraction. Results of experiments on real sequences illustrate that the approach is effective.

1 Introduction

An effective approach to image sequence analysis is the use of layered representations [5]. Sequences are modelled by a set of 2-D layers which are in motion and are at different depths from the camera, where the 'motions' are within the 2-D plane and the 'depths' define the ordering of the layers. Each frame can then be 'reconstructed' by superimposing the layers in order following application of the appropriate motion, in much the same way as cel animations are created. Although somewhat simplistic, layering is able to capture two key characteristics: temporal redundancy as objects appear and reappear in successive frames across the sequence; and the occlusion relationships amongst objects due to depth ordering. This has made it particularly useful for compression applications [5, 6], although it is also likely to be useful in applications such as augmented reality.

Two related tasks need to be tackled when attempting to extract a layered representation. The first is the motion segmentation problem: regions moving with distinct motions need to be accurately identified and tracked. Second, the 2-D motions between these regions and a reference frame need to be determined, ie they need to be aligned with respect to a common frame. Without user interaction or *a priori* knowledge, both of these tasks are difficult and it has proved hard to design a robust algorithm for automatically extracting layers. In some respects this might be considered surprising, given that one might anticipate that the temporal redundancy amongst the frames would make the segmentation task considerably easier than that for a single image. Of course the problem with

BMVC 2001 doi:10.5244/C.15.15

this is that it relies on being able to compensate for the motion between the frames and by implication on gaining reliable estimates of the motion of distinct regions. And it is in this that the crux of the problem lies - to obtain robust motion estimates we require a reliable segmentation of the regions **and vice versa**. In other words, to build robust layered representations we need to come up with effective methods for *fusing* the spatial information within the frames with the motion information across the frames.

The work we describe here tackles this issue head on. Our vehicle for doing this is the *Moving Object Graph* (MOG), a representation of moving regions based on local motion segmentations, ie the graph links together constituent sub-regions of larger regions corresponding to moving objects. Nodes in the graphs are associated with the sub-regions, each of which has its own motion and spatial support, where the latter is either some pre-defined base support for 'interior nodes' or a definition of part of the larger region boundary for 'exterior nodes'. Our motivation for using such a representation is twofold. First, it provides flexibility in terms of the type of region motion that can be accommodated, not relying on fixed parametric forms as has been used by others [5] and potentially able to deal with complex non-rigid or articulated motion. Secondly, from an analysis point of view, it enables us to incorporate both local and global constraints within a coherent framework, allowing local decisions to be influenced by global properties and vice versa, both spatially and temporally. The key point here is that the 'basic elements' of the representation are not pixels themselves, but sub-regions of pixels satisfying a simple local motion model which, being relatively few in number, can be efficiently integrated and updated using global and local constraints. Similar arguments have been made by others using region-based representations, notably Ju *et al* [8] and Szeliski and Shum [7].

We build MOGs by first fusing colour segmentations with block based motion estimates using the integrated motion assignment and depth ordering technique previously reported in [9, 3]. This yields the local sub-regions discussed above and defines the set of nodes in the MOG for each frame. Nodes are then linked within each MOG according to spatial, motion and depth compatibility to give sub-graphs (*MOG components*), each corresponding to a distinct moving region. These components are then linked across the sequence and their constituent motions used to align them with a common frame in order to generate their respective layers. As discussed above, the linking within and between the MOGs enables both spatial and temporal consistency to be imposed and hence correction of erroneous local segmentations, leading to a robust layering algorithm. In the following we first outline the layer and MOG models, and then describe the algorithms for extracting MOGs and building layers. Results of experiments on real sequences are then presented.

2 Layers and Moving Object Graphs

For the layer representation we use a non-transparent version of that originally described by Wang and Adelson [5]: an image sequence is assumed to be a superposition of warped depth ordered layers, such that at each pixel an alpha map 'selects' an intensity from a texture map associated with one of the layers. For each frame, the warp is defined by a spatial transformation derived from the accumulated motion with respect to a reference frame. Let $L^i(\boldsymbol{\xi})$, $0 \leq i < N$, denote the texture maps of N layers and let $\alpha^i(\boldsymbol{\xi}) = 0, 1$ be their associated binary alpha maps, where layer i is 'closer' to the viewer than layer

$i-1$ and $\boldsymbol{\xi}$ is the spatial coordinate vector. Pixels in the kth frame are then assumed to be generated by successively superimposing the texture maps as defined by the recursion

$$I_k^i(\boldsymbol{\xi}) = (1-\alpha_k^i(\boldsymbol{\xi}))I_k^{i-1}(\boldsymbol{\xi}) + \alpha_k^i(\boldsymbol{\xi})L_k^i(\boldsymbol{\xi}) \qquad 0 < i < N \tag{1}$$

with the initial condition $I_k^0(\boldsymbol{\xi}) = L_k^0(\boldsymbol{\xi})$. The frame is then given by the appropriate region of $I_k^{N-1}(\boldsymbol{\xi})$. Here, $L_k^i(\boldsymbol{\xi})$ and $\alpha_k^i(\boldsymbol{\xi})$ are the warped versions of the texture and alpha maps respectively, ie $L_k^i(\boldsymbol{\xi}) = L^i(\mathbf{w}_k^i(\boldsymbol{\xi}))$ and $\alpha_k^i(\boldsymbol{\xi}) = \alpha^i(\mathbf{w}_k^i(\boldsymbol{\xi}))$ where $\mathbf{w}_k^i(\boldsymbol{\xi})$ defines the warp from frame k into the ith layer. Decomposition of a sequence into a layered representation therefore requires generation of the texture and alpha maps, the relative depths of the layers, and the warp fields $\mathbf{w}_k^i(\boldsymbol{\xi})$. We use a 3 step process to do this: (i) identification of the regions on each layer within each frame; (ii) accumulation of the motion and hence spatial transformation between these regions and their corresponding regions in a reference frame (the registration step); and (iii) estimation of the texture and alpha maps for each layer from the registered regions. As noted earlier, we base this process on MOGs extracted from frames in the sequence.

There is one MOG defined for each frame in a sequence and each consists of a set of nodes $n = \{n_i\}$. Associated with each node is an *analysis window* W_i, a *motion sub-region* Λ_i, and a parametric motion $\mathbf{v}_i$, where Λ_i is within the analysis window, ie $\Lambda_i \in W_i$, and we assume here that the motion is translational (although this does not have to be the case - $\mathbf{v}_i$ could define a higher-order parametric motion such as affine). Nodes in the MOG can share a common analysis window (but not a common motion sub-region), ie $W_i = W_j$ for some i and j, and in such cases the set of Λ_i are then contiguous sub-regions of the analysis window, ie if $C_i = \{j | W_j = W_i\}$ then $\cup_{j \in C_i} \Lambda_j = W_i$. The analysis windows are arranged on a regular grid such that each has a set of neighbouring windows, denoted N_i, and links exist in the graph between those nodes with common and neighbouring windows, ie the nodes n_i and n_j are linked if $W_i = Wj$ or $j \in N_i$. The use of the analysis windows is important - they ensure that nodes within the MOG provide a complete representation of the frame, ie they constrain the positioning and density of the nodes.

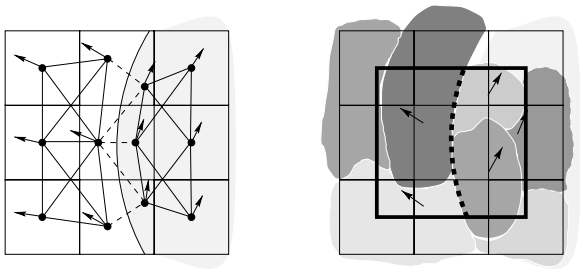
The links between neighbouring nodes have associated binary states s_{ij} defined as:

$$s_{ij} = \begin{cases} 1 & \text{if } n_i \text{ and } n_j \text{ belong to the same moving object} \\ 0 & \text{if } n_i \text{ and } n_j \text{ belong to different moving objects} \end{cases} \tag{2}$$

and thus they indicate two types of link: *intra-object* links ($s_{ij} = 1$) and *cross-boundary links* ($s_{ij} = 0$), which in turn define sub-graphs within the MOG corresponding to distinct moving regions in the frame. We call these sub-graphs *MOG components* and they are defined by a set of *interior nodes*, connecting solely to nodes of the same moving region via intra-object links, and a set of *exterior nodes*, connecting to both 'within region' nodes and the exterior nodes of other moving regions via intra- and cross-boundary links, respectively. Attached to each link is also a relative depth ordering indicating whether the linked nodes are at the same depth or whether one corresponds to an object closer to the camera. We constrain the model such that nodes belonging to the same object are at the same depth (as dictated by our layering model) and thus $s_{ij} = 1$ also indicates equivalent depth.

The states also indicate the presence or not of consistency between node properties, ie if $s_{ij} = 1$ it implies that the motion sub-regions and the motions associated with nodes n_i and n_j are consistent with a common moving object. Specifically, if $s_{ij} = 1$ then

$$W_i \cap W_j \cap \Lambda_i = W_i \cap W_j \cap \Lambda_j \tag{3}$$



(a) (b)

Figure 1: Examples of (a) a moving object graph and (b) a local motion assignment.

and the motions v_i and v_j are compatible with respect to a suitable measure based on similarity or a local parametric form, for example. Conversely, $s_{ij} = 0$ implies that $\Lambda_i \cap \Lambda_j = \emptyset$ and that the motions are incompatible.

In the work described here we have used a simple form of MOG in which the analysis windows are square blocks spaced at regular intervals so as to give 50% overlap between them. An example showing the nodes, links, and associated sub-regions and motions for a neighbourhood containing adjacent moving objects is illustrated in Fig. 1a. The non-overlapping regions of the analysis windows are shown, the intra-object links are shown in bold and the cross-boundary links are dashed. In the next section we describe how we extract such a MOG from a pair of adjacent frames. Note however that the framework could be readily extended by using alternative analysis windows. For example, an obvious extension would be to use variable size windows, giving rise to a multiresolution form of MOG which would be capable of adapting to scale differences within and across frames.

3 Extracting Moving Object Graphs

Extracting the MOG for a frame involves two main steps. First, the motion sub-regions Λ_i are identified within each of the analysis windows. This defines the number and spatial position of the nodes in the MOG. Second, the intra-object and cross-boundary links are formed between the nodes, thus defining the distinct MOG components. We achieve the former by fusing a colour segmentation with block motion estimates using the integrated motion assignment and depth-ordering technique described in [9, 3] and the latter by a local linking process combined with a global constraint mechanism based on an adaptation of the normalised cut algorithm [4].

3.1 Motion Assignment and Depth Ordering

We start with a colour segmentation of the frame of interest and block motion estimates between it and an adjacent frame. The main criterion for the segmentation is that the region boundaries should be coincident with the motion boundaries in the frame and specifically, that the latter should be a subset of the former, ie we make the reasonable assumption that regions of different colour may well be moving together but not vice versa (which is rare). Thus a degree of over-fitting of the segmentation is acceptable and preferred to under-fitting. We have found that the colour segmentation algorithm devel-

oped by Sinclair [2] gives good overall results in this respect. This combines a colour edge extraction process with region growing using seeds at peaks in the Voronoi image derived from the edge map. To help maintain consistency across the sequence we propagated the seeds between frames. The motion estimates were obtained using normalised correlations applied within the analysis windows W_i, applied with a Hamming weighting and implemented via the frequency domain as described in [1].

We determine the sub-regions Λ_i by assigning motions to the colour regions within each analysis window, with the restriction that there are at most two motion sub-regions within each analysis window, ie that a window either straddles a motion boundary between two moving objects or is enclosed within a moving object. This restriction enables a tractable assignment process and in fact is not unreasonable for a suitable size of window (we used 32×32 pixels). Moreover we can deal with the rare case of a window containing three motions by a separate merging process as discussed below. An example is illustrated in Fig. 1b, which shows the motion classification amongst the constituent colour regions of the central analysis window (shown in bold), where the class is indicated by the motion vectors and the motion boundary is shown dotted.

Full details of the motion assignment technique we use can be found in [9, 3]. Essentially, this classifies the colour regions into distinct motion classes based on a motion-compensated difference (MCD) measure which takes into account support for a given motion from both interior and boundary pixels, where the latter is based on an explicit model of boundary ownership and kinetic occlusion. The key advantage of the method is that it is able to deal effectively with regions having low intensity variation for which comparison between motions is problematical since many motions are likely to fit the data equally well. Moreover, since it models occlusions explicitly, the depth relationships amongst the segmented motion regions are obtained automatically as part of the classification process. We also employ a post merging process which combines motion assignments within neighbouring analysis windows in order to correct isolated errors and allow detection of windows containing more than two motions. Finally we employ the *partial correlation* technique to improve motion estimates assigned to the regions, in which knowledge of boundary ownership obtained from the depth ordering is used to reduce bias when estimating the motion of occluded regions. Full details can be found in [3].

3.2 MOG Components

The result of the above analysis is the classification of one or more motion sub-regions within each analysis window, each of which is assigned a node within the MOG for the frame. The next step is to form the intra-object and cross-boundary links within the MOG in order to determine the MOG components corresponding to distinct moving regions. We do this by a combination of constraints on the MOG and requiring spatial and motion similarity between nodes within each component. There are two constraints: (i) nodes with the same analysis window cannot belong to the same component; and (ii) the depth relationship amongst nodes within a component must be consistent. The first of these ensures that the motion segmentations within analysis windows which straddle motion boundaries dictate the formation of the MOG components, whilst the second restricts the set of moving objects to be at distinct depths without intersections as dictated by the layer model. The former is motivated by our observation that errors in the motion assignment stage are primarily the mis-classification of multiple motion blocks as single

motion blocks and thus we deal with these separately once an initial set of MOGs has been extracted using the more reliable classification of boundary blocks (see below).

The aim of the linking process is to estimate an optimal set of link states s_{ij}. We base this on a similarity measure between nodes within neighbouring analysis windows. The states are established using a two pass approach: a local process first determines the optimal states based on comparing neighbouring windows; and then a subsequent global process refines the states so as to produce a set of MOG components which are consistent with the two constraints listed above. The local process works by considering each possible pair of adjacent windows in turn and for each pair starts by establishing links between each node within one window and all the nodes in the neighbouring window. Denoting this set of links by L, the locally optimal set of states S^* is then sought such that

$$S^* = \arg\max_{S'} \sum_{i,j \in L} (2s_{ij} - 1) r_{ij} \tag{4}$$

where S' are those sets of states for which nodes within the same analysis window are not implicitly connected via a third node in a neighbouring window (as required by the first constraint above), ie if $s_{ij} = 1$ and $W_j = W_k$ then $s_{ik} = 0$. Here, r_{ij} is a similarity measure between nodes n_i and n_j based on spatial support and motion, taking higher values for greater similarity and vice versa. Thus since $s_{ij} = 1$ for an intra-object link and $s_{ij} = 0$ for a cross-boundary link, eqn (4) promotes state configurations with high similarity between nodes within the same moving region and low similarity between those in different regions.

The similarity measure we use has the form

$$r_{ij} = \exp(-\gamma p_{ij} - (1 - \gamma) m_{ij}) \tag{5}$$

where p_{ij} and m_{ij} are measures of similarity in spatial support and motion, and $0 \leq \gamma \leq 1$ determines the relative weight placed on each component. The spatial support component is defined as

$$p_{ij} = 1 - \frac{|\Lambda_i \cap \Lambda_j|}{|W_i \cap W_j \cap (\Lambda_i \cup \Lambda_j)|} \tag{6}$$

and reflects the model assumption in eqn (3), ie p_{ij} is close to zero if the motion sub-regions overlap significantly, hence increasing the similarity measure. For the motion, the Mahalanobis distance between that estimated at each node is used, ie

$$m_{ij} = (\mathbf{v}_i - \mathbf{v}_j)^T C_{ij}^{-1} (\mathbf{v}_i - \mathbf{v}_j) \tag{7}$$

where C_{ij} is the sum of 2×2 covariance matrices associated with each motion estimate [1], ie m_{ij} indicates the difference in the motions weighted by their directional certainty.

The above local linking yields an initial set of MOG components which may or may not satisfy the two constraints given above. If not, then we employ a global process which attempts to split inconsistent components into consistent ones. For example, in the case of a mis-classification of a multiple motion block as a single motion block, this can lead to two components being joined via the mis-classified block. However, it is also likely that blocks nearby will contain nodes from both components, thus giving a depth inconsistency within the erroneously detected combined MOG. We therefore seek to remove the link via the single motion block and hence give two consistent MOGs.

The splitting process is a top-down diadic scheme in which we first split an inconsistent MOG into two, check for consistency, and then split again if necessary; repeating until all of the MOGs are consistent. Each split is achieved by a minimal cost removal of links to produce two separate MOGs, ie we remove that set of links whose sum of similarity measures is minimum over all sets which yield two separate MOGs. Identifying such sets is computationally demanding and so we use the more efficient 'normalised cut method' developed by Shi and Malik [4] which, although an approximation, gives good results for this particular application. Finally, we assign a depth index to each MOG component within the frame based on a simple tally amongst its exterior nodes.

4 Building Layers

The next stage of the algorithm is to build the motion layers using the MOGs extracted from the frames in the sequence. There are essentially three tasks involved: (i) to link together corresponding MOG components within the sequence; (ii) to register the corresponding components with respect to a reference frame; and (iii) to build the layer texture and alpha maps for each set of corresponding components. Note that each MOG component is therefore regarded as a distinct layer; in contrast to [5] for example, the layers consist of connected sub-regions, not simply regions considered to be at the same depth.

4.1 Temporal MOG Linking

We form temporal links between MOG components in adjacent frames by first linking interior nodes and then propagating the links out to exterior nodes. Since the interior nodes in all frames lie on the same regular grid as defined by the distribution of analysis windows, this linking requires knowledge both of the motion between frames and of the accumulated motion from a reference frame modulo the size of the analysis windows. Essentially, we 'track' the interior nodes extracted in the reference frame by linking together the interior nodes through which their analysis windows pass as illustrated in Fig. 2. The exterior nodes are then linked in a way which is consistent with the interior links across the sequence. As new nodes appear these are added to the reference frame as 'virtual nodes'. Thus at any given time, all of the nodes can be registered with respect to the reference frame, hence enabling the layer corresponding to the linked MOG components to be created.

At this point we also seek temporal consistency by requiring that the motion assignments amongst linked nodes across a small temporal window are consistent and correct any isolated mis-classifications. This is based on applying the motion assignment technique to all frame pairs within the window and allowing reliable assignments to reinforce each other to give the most confident overall assignment. This proves to be effective, particularly since for frames further apart the kinetic occlusion effects are more pronounced and thus motion assignment and depth ordering are more reliable. The caveat is that in our current implementation we assume a purely translational motion model and thus our temporal window needs to be limited in order to ensure that this model is valid for all frame pairs. Note, however, it is the MOG framework based on a relatively small number of linked local regions moving with simple motions which enables us to impose the temporal consistency.

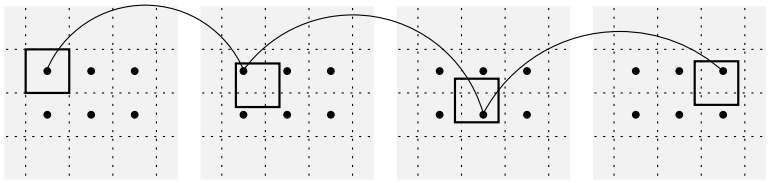


Figure 2: Notional tracking of interior nodes within reference frame by temporal linking of subsequent interior nodes through which they pass.

4.2 Layer Accumulation

To build a layer we need to combine the motion sub-regions within the set of corresponding MOG components. To do this we generate two pixel resolution buffers, one for each layer: a *colour buffer*, to record the colours contributed to each pixel by the sub-regions of each MOG; and a *tally buffer*, to record the tally of contributions made to each pixel. Thus, for a given frame and for each node within a MOG component, pixels within the sub-regions are registered with the buffers using the accumulated motion and then their pixel values recorded in the colour buffer at the appropriate locations. Each corresponding location in the tally buffer is then incremented. Once this has been completed over all frames, the texture map for each layer is generated by computing the median of the colour components at each pixel stored in the colour buffer. Each corresponding alpha map is then generated by setting those locations with a tally above a pre-defined threshold. The tally buffer therefore enables isolated mis-classification of pixels between layers to be discarded, producing more robust layering. For each layer the depth order is simply inherited from the MOG components within the frames, with the constraint that a change of depth order requires assertion over several consecutive frames.

5 Experiments

We have tested the layer extraction algorithm on three sequences: table tennis; a sequence of two gloved hands moving towards each other at different depths; and a sequence of a running Leopard. Example frames are shown in Fig. 3. The initial extraction of motion sub-regions for these frames are shown along the top row, with the non-overlapping portions of the analysis windows indicated by the overlaid grid. The colour segmentation boundaries are shown in white and the motion sub-regions are indicated by light or dark regions. The correct motion assignments and depth orderings have been identified. Note that where the fingertips of the hands meet, the straddling block contains a third motion region corresponding to the background which has been produced by merging neighbouring segmentations. Note also that at this stage there is no consistent labelling between the local segmentations and hence an arbitrary label assignment has been made in each analysis window. The MOGs obtained for these frames are shown in the bottom row of Fig. 3. These show the boundaries of each distinct component (obtained from the sub-regions associated with the exterior nodes), the analysis window grid, and the intra-object links within the MOG.

Finally, Figs. 4 and 5 show the layers extracted for each sequence. In the Hands sequence the background layer starts out as two distinct layers since we require a single

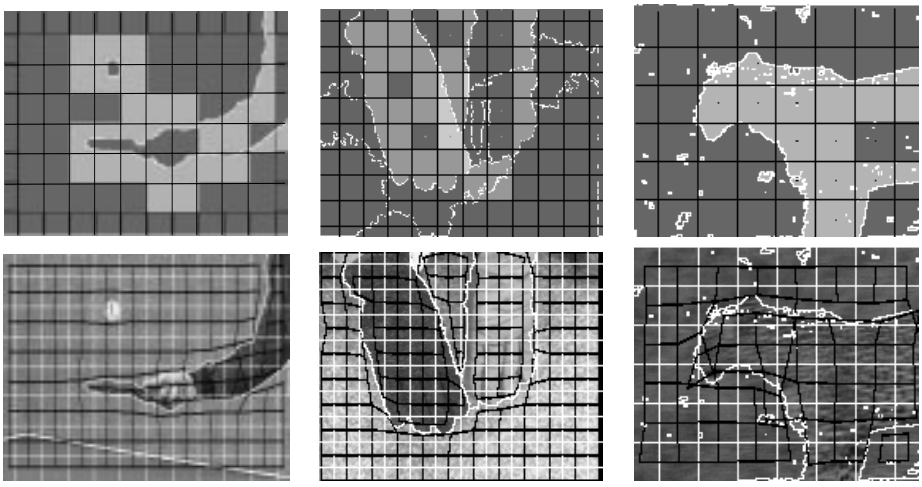


Figure 3: Motion sub-regions (top) and moving object graphs (bottom).

connected region on each layer but these are then merged at a later stage once the Hands have moved apart and the occluded floor underneath becomes visible. Note also that part of the right hand is initially occluded and is then 'built up' within the layer as frames are processed. As an illustration of the effectiveness of the layers extracted, Fig . 6 shows frames from a reconstruction of the sequence from the layers but with the depth order of the hands reversed. The result is surprisingly realistic when viewed as a sequence. The leopard sequence is a particularly difficult example, with large motions between frames and considerable motion blur, as is evident from the slight blurring in the extracted layers. Nevertheless, the shape and key features of the animal in the foreground layer have been successfully built up, despite the fact that only part of the Leopard is visible in any given frame due to camera zoom and pan.

6 Conclusions

We have described a novel approach to motion segmentation and layering. The key contribution of the work is the use of the *moving object graphs* which provide flexibility and robustness while being computationally tractable. Notably it provides us with an effective framework within which to incorporate both spatial and temporal constraints in an efficient and coherent manner. Note however that the extraction of the MOG representations relies significantly on the use of the robust motion assignment method and in particular its ability to simultaneously estimate relative depth order. We are now looking at extending the use of the MOG approach to deal with non-rigid or locally rigid motions, and to generating the representations within a more formal MRF-type framework.

References

[1] A. D. Calway, S. A. Kruger, and D. Tweed. Motion estimation using adaptive correlation and local directional smoothing. In *Proc IEEE International Conference on Image Processing*, pages 614–618, 1998.

Figure 4: Layers extracted from the Table Tennis sequence

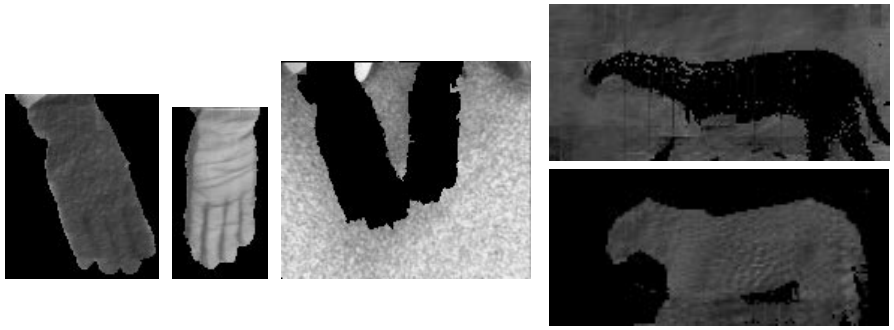


Figure 5: Layers extracted from the Hands and Leopard sequences

Figure 6: Reconstructed Hands with reversed depth order

[2] D.Sinclair. Voronoi seeded colour image segmentation. Technical Report 1999.3, AT&T Labs Cambridge, 1999.

[3] D.S.Tweed and A.D.Calway. Integrated segmentation and depth ordering of motion layers in image sequences. In *Proceedings British Machine Vision Conference*, pages 322–331, 2000.

[4] J.Shi and J.Malik. Motion segmentation and tracking using normalised cuts. In *Proceedings International Conference on Computer Vision*, 1998.

[5] J.Wang and E.H.Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, 1994.

[6] M.C.Lee, W.Chen, C.B.Lin, C.Gu, T.Markoc, S.I.Zabinsky, and R.Szeliski. A layered video object coding system using sprite and affine motion model. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(1):130–144, 1997.

[7] R.Szeliski and H.Shum. Motion estimation with quadtree splines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1199–1211, 1996.

[8] S.Ju, M.J.Black, and A.D.Jepson. Skin and bones: multi-layer, locally affine, optical flow and regularization with transparency. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 307–314, 1996.

[9] D. Tweed and A. Calway. Motion segmentation based on integrated region layering and motion assignment. In *Proceedings Asian Conference on Computer Vision*, pages 1002–1007, 2000.