Dept. of Computer Science
University College London

GV16 - Thesis Literature Review

Tracking powered by Superpixels

Author:
Ahmad Humayun

Supervisor:
Dr. Gabriel Brostow

June 7, 2010

Tracking powered by Superpixels

by

Ahmad Humayun

Submitted for GV16 Literature Review
Research Methods and Reading

Abstract

Motion estimation and segmentation are two critical problems in low-level computer vision. They help reduce raw pixel data to more consumable forms to be used in a wide variety of applications. This thesis is motivated by the lack of a unifying framework for these two methods despite the vast range of schemes available. The aim of this research is two fold: explore the techniques available for over-segmentation and optical flow; and combine them to give a sum that is greater than its parts. The thesis, in its current form, offers a review of relevant literature, and a proposal to combine optical flow and segmentation in an iterative scheme and an algorithm suitability framework.

Chapter 1

Introduction

Pixels, the models we use to represent visual data, are simply a sampling of the visual landscape. Since this landscape is perceived through impressions from photons reflected (or generated) from physical objects, there needs to be some spatial relation between these pixels. In short, pixels next to each other are not part of random process. On further thought, the same reason dictates, if we collect all possible images of the world, the set of pixels will only form a small portion of the space that all random combinations of pixels can possibly occupy. Since a sequence of images of a dynamic scene moves in the same landscape, there is not only a spatial relation between these pixels but also a temporal relation between these stacks of images.

If pixels are looked at as disparate objects, they just amount to a large volume of un-structured data. For most applications, this data can be reduced and still give some higher level reasoning. This partly true due to the spatial and temporal relation in pixels from natural scenes. This is the motivation for most computer vision algorithm. Its critical to note that a lot of current research in the field builds on top of previously devised low-level processing which help reduce pixels into more meaningful forms. Segmentation and optical flow are examples of such low-level algorithms.

To achieve the goal of some higher level understanding, segmentation and/or calculation of temporal motion play a critical role in most vision algorithms. They continuously find use in diversified topics such as multiple-object tracking, object based video compression, machine vision, scene interpretation, and film post-production.

As explained above, they help make data more tractable and meaningful. To fully realise the usefulness of these techniques, it is critical to understand that both segmentation and optical flow are deeply related problems. This intricate relationship is, by no means, a recent discovery, rather has been well-researched under the "motion segmentation" banner. The techniques developed through this thesis would exploit this relationship.

The last three decades have seen a large corpus of work addressing both the issues surrounding segmentation and optical flow (or tracking). A significant number of technical papers around these topics are still published in major computer vision conferences ($\approx 20\%$ papers in CVPR 2010 are either in "Segmentation and grouping" or "Motion and tracking" domain). This shows that the problem of segmentation and motion remain largely unsolved. Techniques, in both these subjects, don't usually perform well in all instances. They tend to fail in some scenes while performing flawlessly in others. Given these large number of algorithms, a framework that could learn to choose the best technique for applying to each portion of the scene, would be extremely useful. This learning-based approach to motion analysis is just beginning to be explored in literature.

Before we can delve into a discussion about the domains of segmentation and motion analysis, we will discuss the goals of this thesis and develop an understanding of the related key concepts.

1.1 Goals

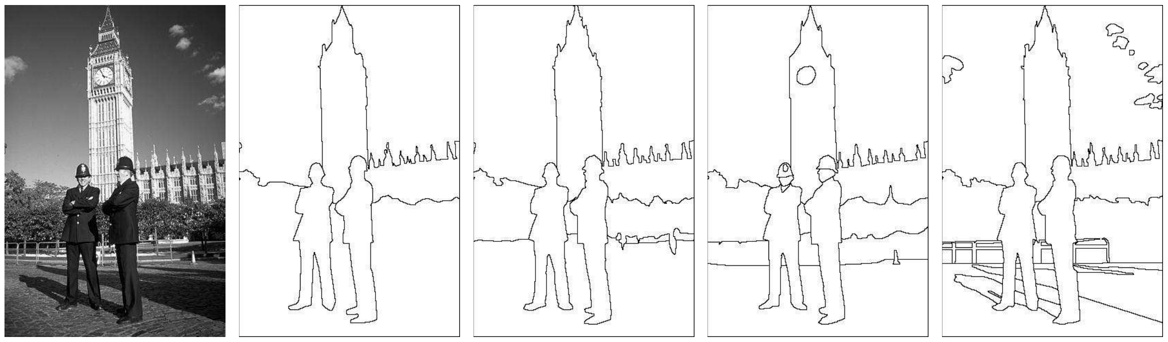
The goal of this research is to explore a new framework for motion segmentation and how current techniques, in this domain, may be improved in a learning-based approach. This thesis will aim to present a new iterative scheme for computing both segmentation (see 1.2) and optical flow (see 1.3) in a way which improves the results produced by both. Since thousands of algorithms already exist for segmentation, optical flow and motion segmentation (see 1.4), developing a new approach might not achieve state-of-art results. To contribute a novel approach, developing a scheme which uses existing techniques, selectively, would be one of the keys aims of this thesis. For this objective, the thesis will aim to develop a learning approach (see 1.5) which, intelligently, applies different off-the-shelf tracking methods.

1.2 Segmentation

Segmentation, simply put, is the division of a visual scene into semantically meaningful portions. To be marked as a segment (separating it from other segments), it should have some physical interpretation. Since different people have different interpretations of grouping, it should be clear why, even for humans, there is no single answer for segmentation. Figure 1.1, from the Berkeley segmentation database (Martin et al. [15]), shows segmentation results performed by 4 different individuals. It is clear that some people perceive individual shadows and clouds belonging to different segments, whereas others only mark out high level objects (in this case buildings, people, and the tree-line).

It would be premature to argue that one set of segmentations is better than others since the answer is highly application specific. Some applications, like Photoshop's bucket tool, might require segmenting every object with a reasonable boundary. Whereas, other applications, like an object recognition system, might just require single segments belonging to complete independent physical entities (like a whole car rather than its individual components). Also applications might require different quality of segmentations in different regions of the scene. It is essential for a Mars rover robot to be able to segment different boulders on the ground to explore a landscape, while another visual sensor on a weather station might be more concerned with the successful segmentation of the clouds in the sky.

Work on general-purpose segmentation in image processing is impeded by this ambiguity in "correct" segmentation. Nonetheless, segmentation can be, more formally, defined as follows: an image I is divided into K segments by the index map $S = \{s_i | s_i \in \{1 \cdots K\}\}$ where i denotes the pixel index for pixel p_i, and $s_i = k$ indicates that i-th pixel belongs to the k-th segment. This should be true for $\forall p_i \in I$. As explained in Section 2.1, some segmentation algorithms assign a single pixel to multiple



(a) Original im. (b) Human segmentation 1 (c) Human segmentation 2 (d) Human segmentation 3 (e) Human segmentation 4

Figure 1.1: Variability in human segmentation. The segmentations are ordered in increasing number of segments. Taken from Martin et al. [15]

segments to tackle cases of occlusion.

The problem of segmentation can be considered hard for the following reasons:

1. The absence of a definitive answer for segmentation (across all applications) leads to problems in creating general-purpose segmentation algorithms. This has caused many segmentation schemes to resort to over-segmentation (see Section 2.1).
2. Lack of colour or texture variability in adjacent objects misleads most segmentation schemes.
3. Segmentation from a single image might not offer enough cues for reasonable segmentation.

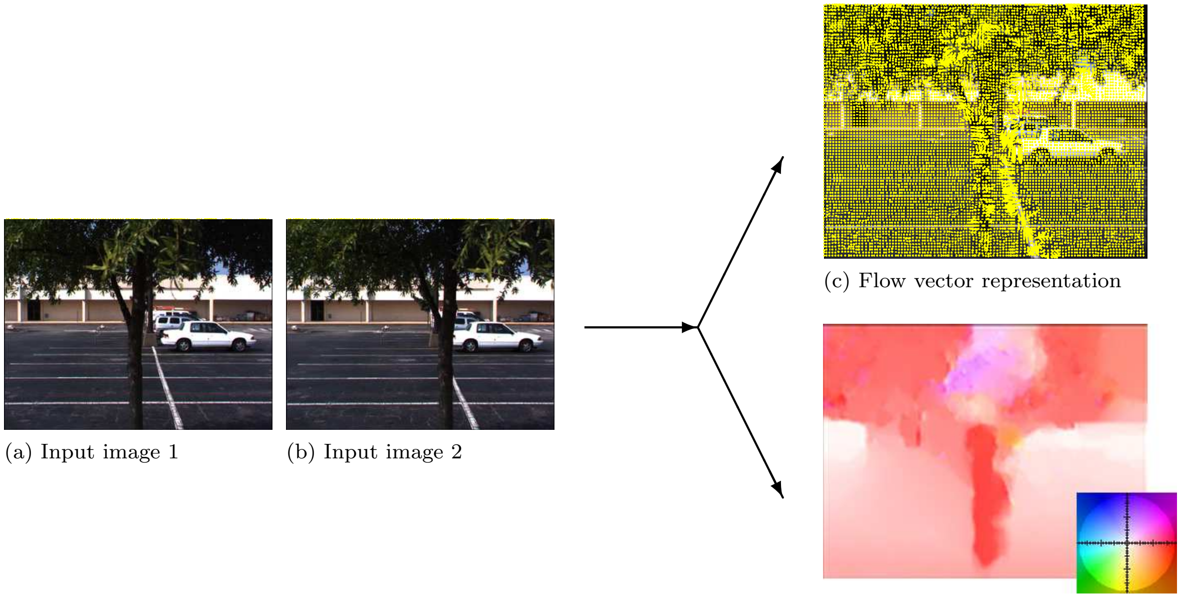
The last point raises the obvious question that can multiple related images help improve segmentation of a scene. The first step in such a process would be to establish correspondence between regions of multiple images. This leads us to our discussion on optical flow.

1.3 Optical Flow

The problem of optical flow helps establish the direction of motion in two or more images. Most of the earlier techniques computed these "motion vectors" for each individual pixel. Algorithms proposed later exploited the constancy of motion across planar segments of images. Optical flow can be thought of a specialised segmentation problem itself (see p. 12, Ross [24]) - segmenting a set of images into regions of motion.

Figure 1.2 shows two representations used to display optical flow results. Figure 1.2d shows the representation in which the different flow vectors are painted using a colour wheel.

Formally, optical flow is usually defined as follows: given two images I_1, I_2 and regions in them r_i^1, r_j^2, optical flow defines two things: correspondence in regions i.e. $r_x^1 \equiv r_y^2$, and the motion parameters Θ_{xy}^{12} that transform r_x^1 to r_y^2. As explained above, earlier optical flow schemes operated on individual


(a) Input image 1

(b) Input image 2

(c) Flow vector representation

(d) Color encoding flow representation

Figure 1.2: Shows the two common optical flow representations

pixels, in which case r_i^1 would refer to individual pixels in image 1, and the motion model Θ could only be translational. In frameworks which deal with regions/segments, r_i^1 would refer to a collection of contiguous pixels and the motion model could be anything from pure translational to projective.

Most segmentation and optical flow schemes are riddled with the same set of problems. The most discussed/researched problem out of them is the problem of finding accurate segmentation or optical flow for regions close to object boundaries. Most techniques, in both domains, model this problem explicitly in fear of failing on these contentious boundaries. This possibility of failure is largely due to assumptions in respective techniques in order to make them tractable: many optical flow techniques demand that pixels/regions don't change colour as they move; similarly algorithms for segmentation on monocular images suppose that object discontinuities usually coincide with colour/texture discontinuities.

Shi and Malik [27] point out that early approaches for optical flow took clues from discontinuities only in a post-processing step. This proved disadvantageous for mainly the following reasons:

1. Large planar regions with one-dimensional or no texture are notorious for simple optical flow techniques. Discontinuities carry key to solving these regions correctly.

2. To solve (1), smoothing constraints were proposed to interpolate flow fields. But to incorporate some smoothness constraint, the image needs to be segmented before hand to avoid smoothing over discontinuities!

This provokes the thought that the optical flow and segmentation are intimate problems. This necessitates a unified framework that uses both sets of information in parallel.

1.4 Motion segmentation

It is worthwhile noting what constraints segmentation and optical flow techniques impose on themselves to make their problems manageable. An important supposition in most segmentation schemes is the smoothness of colour and texture across planar surfaces in the world. This relates to a common presumption in optical flow schemes where smoothness of motion needs to coincide with smoothness of colour and texture. Similarly segmentation schemes suppose object boundaries to fall at feature discontinuities; likewise, optical flow schemes also expect motion discontinuities with drastic change in features. Apart from these, many other parallels can be drawn between these two problems. The fact that they aim to deal with a similar problem, i.e. segmentation of physical objects, makes them highly connected. It is no surprise that many techniques in both domains are also plagued by the same underlying problems.

Nonetheless, there are also clear differences in the formulation of the two problems. Segmentation mainly looks at spatial discontinuities in image features, whereas optical flow seeks temporal discontinuities in features. Being able to work on different, compatible, sets of information suggests that both can participate in a common framework, where each helps produce a better output for the other.

Many motion segmentation techniques suggested in literature (see 2.3) fuse these two techniques. They aim not only to produce motion vectors for image features, but also dissect frames into regions. In literature, there are two main categories of segmentations in motion segmentation schemes. One common scheme is to create segmentations independently across frames. These segmentations would show some similarity in motion and image features. The second scheme is more semantically meaningful, where segments in a frame are corresponded with segments in other frames. This technique aims to produce whole temporally consistent segments.

Like its component techniques, motion segmentation is far from being a perfect technique. To improve such a corpus of techniques, one possible solution is turn to learning-based approaches.

1.5 Algorithm suitability (learning approach)

Algorithm suitability, the learned selection of an algorithm according to the given data, is a well discussed problem in the machine learning domain. It aims to provide, a single, more accurate solution given a number of *experts* (classifiers) for solving a particular question. In this thesis we are more interested in the problem when the classifiers themselves are individual algorithms. This allows us to turn the problem of choosing an ideal classifier to a problem of selecting the right algorithm.

Hence, algorithm suitability can be considered as follows. Given a feature set $x \in \mathbb{R}^n$, we need to produce a label y, such that $f : \mathbb{R}^n \to y$. In the case of algorithm suitability, y is the choice of algorithm and x is the feature set used to produce it.

In the cases we will discuss, x would usually be a set of spatio-temporal features, and $y \in Y$ where Y is the set of all available algorithms. This specialised classification problem is of particular interest in this thesis since there is lack of available schemes to intelligently choose algorithms in the domain of tracking (and motion segmentation). As we approach the end of this thesis, we will propose a novel scheme for algorithm suitability for tracking (or mothion segmentation).

1.6 Organisation

The thesis is organised into 4 chapters: this introduction (1); the related work chapter (2); our approach (3); and conclusion (4). The thesis, in its current form, concentrates more on the related work chapter, which touches literature in four main subjects: (1) segmentation (2.1); (2) optical flow (2.2); (3) motion segmentation (2.3); (4) algorithm suitability (2.4). The literature discussed revolves around segmentation, flow computation and techniques which unify or improve the two methods.

Chapter 2

Related Work

This chapter gives an overview of the literature in the area of segmentation (Section 2.1), optical flow (Section 2.2), motion segmentation (Section 2.3), and algorithm suitability (Section 2.4. Since each of these areas have huge corpuses of literature, it would be impossible to survey them completely. Apart from discussing seminal works, this chapter will review publications that have found use in *both* segmentation and optical flow. It will also devote a thorough section on motion segmentation techniques and, lastly, algorithm suitability schemes.

More thorough reviews of individual papers, discussed below, can be found at `http://www.cs.ucl.ac.uk/students/A.Humayun/resources/paperreviews.pdf`.

2.1 Segmentation

Most of the schemes in generalised segmentation have, over the years, moved from estimating exact solutions to finding over-segmentations. The reason is two-fold. As discussed in Section 1.2, the exact solution to segmentation is not always defined. Secondly, over-segmentation schemes are more useful for high-level algorithms, which can pick and choose segments according to its needs. In this section we aim to discuss segmentation schemes which rely on colour and texture cues, without any *prior* information for the segments we seek.

Graph based approaches are an interesting formulation of the problem, as it allows segmentation to be dealt with various graph-based techniques like *MRF*s and *Graph-cuts*. Felzenszwalb and Huttenlocher [6] proposes one such technique, which starts with each pixel being a segment (Figure 2.1c). From here it merges segments which satisfy this constraint:

$$Dif(C_1, C_2) < \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2)) \;= MInt(C_1, C_2)$$

where $Int(C)$ is the maximum weight connecting the segment, $Dif(C_1, C_2)$ is the minimum weighted edge between two segments, and $\tau(C) = k/|C|$ provides support for mergers when segment sizes ($|C|$) are small. Each step involves sorting un-connected edge strengths, and making mergers across smallest weighted edges. This sheme is repeated until it cannot make any more merges.

Shi and Malik [28] well-known method also employs a graphical model, but instead of growing regions, it starts with every pixel being part of a single large region (Figure 2.1b). Then it divides the image by successive *normalised cuts* until a cut-value threshold is reached. The *normalised cuts* criteria is based on the familiar *minimum cut* problem from algorithmic graph theory. *Min-cut* attempts to divide a graph into two unconnected subgraphs by removing edges such that the combined weight of the removed edges is minimised. "Therefore, if the weight of edges varies proportionally with the similarity between the pixels it joins, finding the min-cut of a graph should separate it into dissimilar regions" [24]:

$$N\text{cut}(A, B) = \frac{\text{cut}(A, B)}{\text{asso}(A, V)} + \frac{\text{cut}(A, B)}{\text{asso}(B, V)}$$

This framework is also used for tracking, as reviewed in Section 2.2.

Yu and Shi [38] provides another graph based approach, which exploits both region and boundary information for segmentation. As compared to the approaches mentioned above, it uses two graphs: (1) pixel graph $G = (V, E)$ where weight on edges encode region cues (intensity, texture, motion similarity); (2) edgel graph $\partial G = (\partial V, \partial E)$ (the graph where nodes become edges) where weight on edges encode contour cues/boundary cues. Consistent partitioning on these two graphs is guaranteed by the coupling of their partitioning membership vectors through edge-node incidence matrices. Such a link through linear operators allows deriving eigendecomposition solutions to a global optimisation criterion. Yu et al. [40] uses this technique to approach segmentation as a top down problem.

Active contour models in segmentation were popularised by Kass et al. [11] which evolves a curve around an object. It uses the following equation:

$$E_{\text{snake}} = \int_0^1 E_{\text{internal}} v(s) + \text{image} v(s) + \text{constraint} v(s) ds$$

where $v(s)$ are the points that define the contour; E_{internal} is the stretching and bending energy in the curve given by the points; E_{image} measures the attraction of features of the to-be-segmented region; and $E_{\text{constraint}}$ measure the external constraint like some shape information, or user applied energy.

Paragios and Deriche [21] expands on active contour models by defining a *Geodesic Active Region* model. Like Yu and Shi [38], this model incorporates both boundary and region information in a unified approach which is minimised in a curve-based framework. To incorporate multiple curves in a single formulation, it uses a level-set approach which helps introduce artificial forces penalising pixels that fall in multiple regions or don't fall in any region at all. The paper argues for its applicability in various frame-partitioning problems like image segmentation and supervised texture segmentation. The precursor to this work also discusses the use of this approach in motion segmentation (discussed in Section 2.3).

An interesting clustering based segmentation method is proposed in Schroeter and Bign [25] which uses a multi-resolution approach. This approach was popularised in optical flow to avoid adding spatial locality as a feature. By initiating clustering at extremely low resolution and propagating the results downwards, spatial locality is enforced at the coarse levels and persists into the higher-resolution pyramid base.

Unlike most of the approaches in segmentation, Zitnick et al. [45] adopts a generative approach for associating up to two segments to each pixel. The paper argues for this approach because the colour of a pixel might be a result of two surfaces in a scene (a scenario often encountered in motion blur). This will be discussed further in Section 2.3.1.

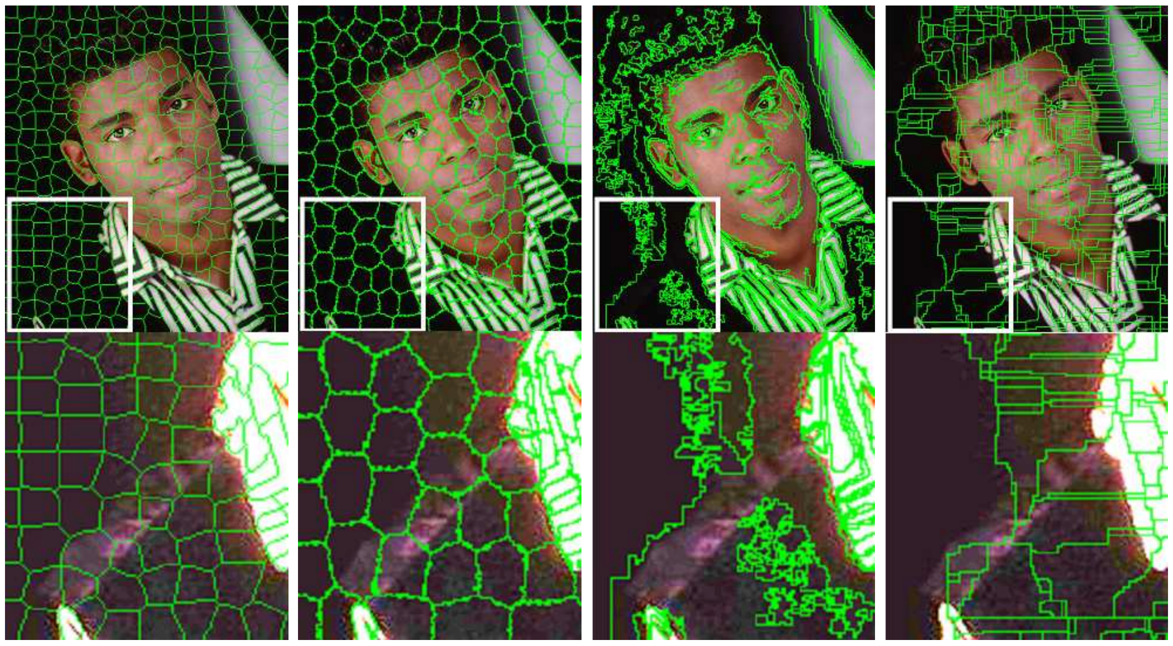

(a) Turbopixels (b) Ncuts (c) Local variation (d) Watershed

Figure 2.1: Shows outputs from different segmentation techniques: 2.1a [13], 2.1b [28, 39] use a compactness constraint; 2.1c [6], 2.1d [33] are general over-segmentation techniques. Taken from Levinshtein et al. [13]

2.1.1 Superpixels with homogenous size

Superpixels is a term popularised by Ren and Malik [23]. These types segments make image data more tractable for high level algorithms while losing little information which were available in individual pixels. Their properties include them being "local, coherent, and which preserve most of the structure necessary for segmentation at the scale of interest" [23]. In most applications, such as motion segmentation, it is useful to have *superpixels* which are approximately of the same shape and size. The schemes described in Shi and Malik [28], Felzenszwalb and Huttenlocher [6], which are discussed above, can be categorised as a *Superpixel* algorithms, which don't constrain segments to be of homogenous shape and size.

Stein and Hebert [31] work on detecting occlusion boundaries from motion, provide an interesting module for image over-segmentation. As a starting point it chooses the P_b (posterior probability) responses used in Martin et al. [16]. This is a good starting point as this posterior probability results from features trained to capture changes in brightness, colour, and texture which coincide with natural boundaries. Using non-local maxima suppression on these P_b responses, it creates an edge map. To keep the size of segments homogenous, it introduces random seeds in the edge map before computing the final segments through the watershed algorithm. Section 2.3.1 discusses how these segments are later used for motion estimation and extraction of occlusion boundaries.

One of the recent *superpixel* approaches is given in Levinshtein et al. [13] which introduces the *Turbopixel* algorithm (Figure 2.1a). It is, basically, a curve evolution geometric flow with a constraint to avoid under-segmentation (compactness constraint). Starting with K (user-specified parameter) seeds, 1 pixel radius curves are defined around each seed. To avoid placing seeds on edges, it moves them in the direction of the image gradient as a function of the image magnitude. The curve is evolved using a level set formulation of a smooth and continuous function Ψ:

$$\Psi^{n+1} = \Psi^{n} - S_I S_B ||\nabla \Psi^{n}|| \Delta t$$

where Δt denotes the single time step in the evolution of the boundary, and $||\Psi^n||$ denotes the normal direction of the curve. $S_I S_B$ are the product of the main speeds which evolves the curve in the direction of the normal. The proximity-based boundary velocity S_B ensures that the curve evolves only with respect to the image features until it gets close to another *superpixel* region. Since, at any time instance, the approach is only concerned with the zero level set of Ψ, it only maintains an accurate representation of Ψ in a narrow band around the curve boundary. This equation is iterated over until no evolution of the curve is possible.

Many other application specific literature introduce their own segmentation schemes. Zitnick et al. [43] uses a colour segmentation as an input to its stereo algorithm. Since they use a *layered* based approach (refer to Section 2.3 for *layers*), this work can be easily applied to a tracking framework. Each segment is initially associated with a single disparity label, which helps in reducing susceptibility to noise. It supposes that disparity discontinuities coincide with colour discontinuities (but also specifies that disparity within a segment need to be smooth but not planar). It is, essentially, a region growing segmentation algorithm which starts by smoothing the image with anisotropic diffusion. It merges neighbouring segments if the euclidean distance between their average colour is within a constant parameter (they use a distance of 6). Segments smaller than 100 pixels are merged with adjacent segments who are closest to their colour, whereas segments larger than 40 pixels in length are broken across their width and height. From a tracking perspective, its interesting to note that these disparities are later smoothed out by re-projecting the segment to all the other frames (by averaging disparities with the overlapping segments in these other frames).

Its follow up work appeared in Zitnick and Kang [42] which devises a stereo method for image based rendering. It updates the technique in [43] for over-segmentation to a K-means approach. After performing the same smoothing step, the image is partitioned into 8×8 pixel blocks. Each block/segment is then modelled with Gaussian for its colour space and spatial extent. Using an iterative K-means algorithm it refines the boundaries using the model developed for each segment.

2.2 Optical Flow

As discussed in Section 1.3, computing motion flow has a lot of uses in many computer vision algorithms. Yet for them to be used as effective tools, they need to resolve motion accurately even at motion boundaries. This section will explore a very small section of literature related to optical flow which either concentrates on motion / intensity discontinuities, or smoothly / linearly varying regions using some special scheme.

One of the earliest approaches for dense motion estimation, Heitz and Bouthemy [8] uses multiple complementary constraints in an attempt to preserve motion at boundaries. The first constraint it uses is on Gradient-based motion. Initially proposed by Horn and Schunck [9], it suggests:

$$\vec{\nabla} f(s).\vec{w}_s + f_t(s) = 0$$

where $f(s)$ is the velocity vector at a particular point, $\vec{\nabla}$ is the spatial image gradient, and f_t is the temporal intensity gradient. This equation shows that only the flow parallel to the spatial image gradient can be recovered if we only rely on local computation. This is the well known *aperture* problem. This was tackled earlier by either supposing smooth velocity variations across the image or invariant velocity in a small neighbourhood. The second edge-based motion constraint is used by thresholding the log-likelihood ratio test based on parameters defined on a surface, where each parameter is tuned to detect edge-locations, orientation and displacement. This paper introduces validation of these constraints using hypothesis testing. This is followed by using image features as observations in a Bayesian estimation process, used to extract motion labels for each portion in the image. The relationship between the observation fields and the labels in this process is specified using an MRF.

It has been argued that the constraints used in the Bayesian method significantly increases the computational complexity of the method. To tackle this, Chang et al. [5] introduces interdependence between the optical flow field and the segmentation map directly through the Bayesian framework. It uses a motion field as a sum of a parametric field and a non-parametric residual field. This helps it conveniently resolve segmentation and optical flow by simply finding the parameters for the parametric field using a least squares solution. The optical flow field is refined by computing the (non-parametric) minimum-norm residual field given the best estimate of the parametric field, under the constraint that the motion field needs to be smooth within each segment. This gives the best estimates for the motion field and segmentation. The segmentation portion of this method will discussed further in Section 2.3.

Shi and Malik [27] introduces a graph based approach to estimate flow fields and motion segmentation (discussed in Section 2.3). It forms a graph $G = (V, E)$ which are pixels connected in its spatiotemporal neighbourhood with weights $(w(i, j))$ denoting the similarity of motion between two pixel nodes. Although only motion weights are used, it mentions that weights can also be based on measures such as colour, brightness, texture, and disparity. It takes the approach of committing motion vectors later in the pipeline, by initially working with just a *motion profile*: the probability distribution of the image velocity at each pixel in the image, which captures both direction and uncertainty:

$$\frac{1}{Z}\exp(-\alpha \mathrm{SSD}[I^t(x_i), I^{t+1}(x_i + u)])$$

where I^t and I^{t+1} are the two image patches and α is the weighting, and Z is the normalisation constant. The weight on the graph is taken as the cross-correlation of the two motion profiles given by P_i and P_j:

$$w(i, j) = \exp(-[1 - \sum_{dx} P_i(dx)P_j(dx)]/\sigma_m^2)$$

where σ_m^2 is the expected variance in the motion profiles. It should be noted that this measure of motion similarity will, indeed, distinguish between two pixels which have exactly the same true motion, but different brightness profiles, which would result in difference in associated motion uncertainties. This will happen if one of the pixels is in a region of constant brightness and another in a region of rich texture. The method handles this in a post-processing step. Eventually the algorithm summarises and solves all motion profiles between each pair of pixels as an eigenvalue problem.

In a related approach, Galun et al. [7] proposes to iteratively improve motion flow estimates by solving systems with increasingly complex motion models. At each level of complexity it chooses seed pixels for computing optical flow using Shi and Malik [27]. It uses the same cross-correlation method to compare motion profiles, after smoothing each profile with a Gaussian. These seed pixels are later associated to individual clusters signifying common motion. A re-estimation step solves for the selected motion models (according to the iterative level) while estimating a common motion for the cluster. As discussed in Section 2.3, this helps in combining motion by supposing that all pixels are strongly associated with a subset of the selected seed pixels.

Following the same idea of employing increasingly complex motion models, Wong and Spetsakis [35] proposes another motion segmentation method for tracking objects on a static background. The tracker, in this technique, needs to be initialised with a small seed window which falls inside the to-be-tracked object in the first frame (it proposes a 10×10 pixel window). Using Least SSD, it attempts to estimate the initial optical flow vectors:

$$\text{LSSD}(u, v) = \min_{(u,v)} \sum_{x,y \in S} (I_{t-1}(y+u, x+v) - I_t(y,x))^2$$

where S is the seed window. Since the seed window is small, the search window is set to be relatively large, going from $-30 \cdots 30$ pixels. To refine the initial estimate of (u, v), it aligns I_{t-1} to I_t (using the initial (u, v)) and attempts to compute the sub-pixel optical flow. This is done by moving the window successively in the 8 surrounding directions and finding the location which minimises the SSD. The motion segmentation technique that follows is discussed in Section 2.3.

2.3 Motion Segmentation

Motion segmentation techniques combine the *segmentation* and *optical flow* paradigms to solve both problems in a unified way. As discussed in Section 1.4, there are two main approaches to *segmentation* in motion segmentation techniques. In this section we will discuss a small section of the large volume of literature available on this topic.

Black and Jepson [3] proposed one of the earliest methods to constrain motion to regions using the brightness information. The solution given can be divided into two stages: early processing and medium-level processing. In this two-stage process, it attempts to estimate optical flow through motion of planar regions and local deformations. These deformations are allowed in the model since the assumption of planarity is likely to be violated in any natural scene. The segmentation, coupled with this method, is done on brightness values to constrain motion to planar regions, as initially proposed in Black [2]. It uses an analog spatial outlier process to define discontinuity between pixels - also defined is a penalty term which needs to be paid with increasing discontinuity. All these steps are performed to eventually minimise an objective function with a data term and spatial coherence term. The first stage of the method estimates a coarse fit to the parametric model and evaluates a set of parameters for each region. It consists of two low-level processes: a process that smooths image brightness while checking for discontinuities; and one that estimates motion. The second medium-level processing stage refines the initial fit of the parameters with “standard area-based regression approaches”. The main aim of this stage is to collate the low-level information from the first stage by connecting piecewise smooth brightness regions and estimating their motion. This is done in three steps: (1) fit a translational/affine/planar model which best captures motion in a region, (2) using this model, warp the regions into alignment, so a Gradient-based optical flow method can help in refining the initial parametric model, (3) allow deformation of low-level patches to improve the motion estimate of each planar patch.

As introduced in Section 2.2, Chang et al. [5] takes a slightly different approach to motion segmentation - by finding a parametric field, that optimises the motion field and segmentation. This is done using a least squares solution. After refining the estimates for the flow field, the segmentation field is also improved to give the minimum-norm residual field using Gibbsian priors. The least squares estimates of the mapping parameters Φ for each segment is computed in closed-form given the MAP estimate:

$$(\hat{u}, \hat{v}, \hat{S}) = \max_{u,v,S} P(I_t|u, v, s, I_{t-1})P(u, v|s, I_{t-1})P(s|I_{t-1})$$

where (u, v) are motion field vectors, I_t (current frame), I_{t-1} (search frame) are the two frames, and s is the segmentation field. The conditional PDF $P(I_t|u, v, s, I_{t-1})$ quantifies how well the motion and segmentation estimates fit the given frames. This PDF is modelled by a Gibbs distribution. $P(u, v|s, I_{t-1})$ is also modelled by a Gibbs distribution with a potential function which aims to minimise the deviation of the motion field (u, v) from the parametric motion (u_p, v_p). The third term $P(s|I_{t-1})$, the priori probability of the segmentation, also follows a Gibbs distribution to discourage formation of small, isolated regions. The method also proposes dense representation of the residual field for improved motion segmentation.

The graph based approach due to Shi and Malik [27] aims at finding motion segments which minimise the normalised cut, as defined in 2.1. Once the graph is constructed (as discussed in Section 2.2), normalised cuts across the graph will give spatiotemporal volumes corresponding to different moving objects. This technique segments the scene since normalised cuts not only reflect similarity within a partition but also dissimilarity across partitions. Combining the weights $w(i, j)$ in a matrix $\mathbf{W}$, and a diagonal matrix $\mathbf{W}$ where $\mathbf{D}(i, i) = \sum_j w(i, j)$, the method solves the generalised eigensystem $(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda\mathbf{D}\mathbf{y}$ for the smallest eigenvalues. The graph is then partitioned by Ncuts with the eigenvector belonging to the second smallest eigenvalue. The segment is only used if it is stable (by checking the cut cost). As a result, time slices of the output segments indicate corresponding groups across time.

Since [27] groups pixels based on the affinity of the motion profile, a local measurement, it ignores global constraints and appears unstable in noisy sequences. An approach based on *Geodesic Active Region* due to Paragios and Deriche [20] tackles this problem by incorporating a *Visual Consistency Module* which tries to optimise the segmentation map globally (while keeping track of motion). The main theme of the paper is simultaneous tracking of multiple non-rigid objects. A more generalised formulation of this approach can be found in Paragios and Deriche [21] (discussed in Section 2.1). The method, at its heart, has a curve-based objective function formed with boundary and region-based terms. This final objective function is minimised using gradient descent methods. The boundary terms aim to find a minimal length contour attracted to region boundaries. On the other hand, region-based terms aim to maximise the quality of the segmentation map. Intuitively, the initially proposed curves are propagated toward the best partition under the influence of boundary, intensity and motion-based forces. The method assumes object motion can be described using global affine model $A(x, y)$.

This technique also incorporates a motion detection module which uses the difference frame to create parametric (Gaussian or Laplacian) distributions which can be used to model static and mobile pixels. The parameter in these distributions are estimated using the Maximum likelihood principle.

Also particular to this method is the intensity segmentation module - which moves the curve in the direction which creates interior regions with desirable intensity properties.

Another method which considers an affine motion model was proposed in Wong and Spetsakis [35]. Its initial estimation of the flow field was discussed in Section 2.2, which resulted in sub-pixel estimates of (u, v). This helps align the image segments I_{t-1} and I_t. Since the computation of the affine flow can be costly, the technique uses the initial translational estimates to identify areas where to compute an affine model. This model is then built using a differential approach which computes the standard least squares for the following equation:

$$\mathrm{SSD}_{\mathrm{affine}} = \sum_{\text{all } x,y} {}^{(t-1)}I_{\mathrm{track}} \cdot \left[I_{t-1}(x,y) - I_t(x,y) + I_{t-1,x}(x,y) \begin{bmatrix} u_0 \\ u_x \\ u_y \end{bmatrix} + I_{t-1,y}(x,y) \begin{bmatrix} v_0 \\ v_x \\ v_y \end{bmatrix} \right]^2$$

where $\begin{bmatrix} u_x & u_y & u_0 \\ v_x & v_y & v_0 \end{bmatrix}$ are the affine parameters, $I_{t-1,x}$ and $I_{t-1,y}$ are the image derivatives in the x and y direction, and ${}^{(t-1)}I_{\mathrm{track}}$ represents the region on which the affine optical flow is being computed. The affine parameters are computed by minimising $\mathrm{SSD}_{\mathrm{affine}}$ for all pixels in the region. By aligning all the previous images to the last image using these affine parameters, it segments out the object to-be-tracked by thresholding the pixel-wise SSD. This is a reasonable measure since the SSD of the aligned tracked object would be small. This process is made computationally tractable by aligning just the first and second moments of images (see [35] for details). To segment out the moving object, the threshold is set keeping in mind the camera and motion noise.

As discussed in 2.2, Galun et al. [7] also iteratively improves its motion models in an effort to segment motion. The approach involves iterating over two steps: *clustering* and *re-estimation*; where each iteration is stated as a *level*. Apart from computing the optical flow for each seed pixels in the *clustering* step, it also computes how strongly each pixel is associated to a particular seed. The aim of the *re-estimation* step is to estimate the common motion of each cluster. This coarsening step begins by selecting a subset of the elements from the previous level as seeds, with the constraint that all other elements are strongly associated with (subsets of) these seeds. To aggregate pixels, the motion profile is computed by multiplying all the *child* motion profiles (see [7] for details) - this technique gives sharply peaked motion profiles in textured regions in just 1-2 coarsening steps (note, this scheme doesn't work in uniform regions). These *peaked motion profiles* help accumulate moments of respective seeds. With increasing levels, the aggregates/segments become large and translational motion stops reflecting the true motion. If there are enough constraints available in the *peaked motion profiles*, an affine transformation for the segment can be computed. The paper also describes computing a projective transformation with the fundamental matrix at higher levels. Finally, these motion parameters are combined with intensity cues by using Segmentation by Weighted Aggregation (SWA) (Sharon et al. [26]) to give the output segments.

Stauffer and Grimson [30] also aims to learn patterns of activity in a scene. Their approach is unique amongst the works mentioned here, as it proposes a system which fuses motion information from multiple-cameras (each sensor has some location awareness). Motion segmentation is done using an adaptive background subtraction method where each pixel (process) is modelled as an adaptive mixture of Gaussians, using online approximations to update this model. Each time the parameters of this model are updated, it uses a heuristic to find whether the pixel belongs to a background process or not. Every new pixel X_t is checked against K Gaussian distributions until a match is found (it considers a value within $\mu \pm 2.5\sigma$ of a distribution as a match). If none of the distributions match the current pixel, the least probable distribution is replaced by a new distribution, with a mean of X_t and a high variance and low prior weight $w_{k,t}$. Finally, the Gaussian distributions, having more evidence (prior weight) and less variance (sort distributions by w/σ) are taken as part of the background. Pixel values that do not match any of the background Gaussians, are grouped together using connected components. These connected components are tracked across time using a multiple hypothesis tracker, hence resulting in segments with their motion.

To discuss a subspace method for motion segmentation, we will turn our attention to Zelnik-Manor et al. [41]. This paper tracks using subspace methods applied directly to features pixel intensities. It organises the problem as a multi-body segmentation using a flow field matrix $[U|V]$ (where both U and V are matrices of directional motion vectors of Frames $\times$ Pixels dimensions) giving rise to multi-body factorisation with directional uncertainty. While segmenting $[U|V]$, the decision of grouping together two objects is based on the ranks of the matrix - which exploits the linear dependency of flow fields of a single object. Since the flow-field matrix $[U|V]$ is of rank 1, there exists a set of basis trajectory vectors and a set of basis flow-fields such that they can be factored into two matrices:

$$[U|V] = \underbrace{C}_{\text{flow coefficients}} \underbrace{B}_{\text{flow basis}}$$

In addition, since tracking might not be reliable for all feature points, it introduces directional uncertainty by a weight matrix $[U|V]Q$. Irani and Anandan [10] proves that this results in the covariance-weighted measurement matrix ($[G|H] = [U|V]Q$). Finally, segmenting the entire video into a set of moving objects is now a question of sorting the columns of the covariance-weighted measurement matrix. This is easily done by finding its RREF. Since the rank of $[G|H]$ is considered to be small, RREF is computed on the SVD of $[G|H]$.

The reader is referred to Megret and DeMenthon [17] for a more detailed survey (and taxonomy) of motion segmentation techniques.

2.3.1 Occlusion resolution

Occlusion is a critical concept algorithms have to deal with in motion segmentation. The literature discussed belongs to either occlusion resolution in a *layered* representation or boundary occlusion resolution.

Wang and Adelson [34] uses the term *layers* for moving objects. The method basically gives a method for motion segmentation where each segment becomes a *layer*. Each *layer* is defined using an intensity map and an alpha map (to denote each pixel's transparency). Velocity maps are used to show how *layers* can move in time. Each *layer* is also assigned a depth ordering which follows the rules of compositing. Since optical flow *models* moving objects like "rubber sheets", it argues that this motion assumption breaks down in case of occlusion. It supposes that regions undergoing similar affine motion result from the same plane in the world - and to deal with boundary motion discontinuities, it allows sharp breaks in the flow field using regularisation. These discontinuities are explained by the framework as instances of occlusion. For an initial estimate of motion, optical flow is computed in a local neighbourhood region, as suggested by Bergen et al. [1]. Using these initial estimates of optical flow, it determines a set of affine parameters which are likely to be observed. The scene is then segmented, by an iterative process, which classifies regions of the motion model that provides the best description of the motion withing each region. This technique relies on k-means clustering in affine parameter space. This method was arguably the first to develop the idea of an affine model for flow.

Following a layered based approach, Zitnick et al. [43] (discussed in 2.1.1) employs an interesting matting technique in order to interpolate views from multiple images. Since boundary pixels might receive contributions from foreground and background objects, it argues that the same colour distribution for boundaries in new views might look unnatural. It approaches this problem as an overlap of layers. In locations of depth discontinuities, matting information is computed in the 4 neighbourhood pixels. Within this neighbourhood, foreground and background pixel colour with alpha values are computed using Bayesian image matting. The information recovered for the foreground is used to compute a new boundary layer. This is useful in generating novel views since the boundary layer can be rendered with different levels of opacity.

Zitnick et al. [45, 44] also computes motion segmentation while accounting for matting in overlapping regions (modelled with α). It proposes a matting model where each pixel can belong to two segments (out of the K segments) with a corresponding association of α and $1 - \alpha$:

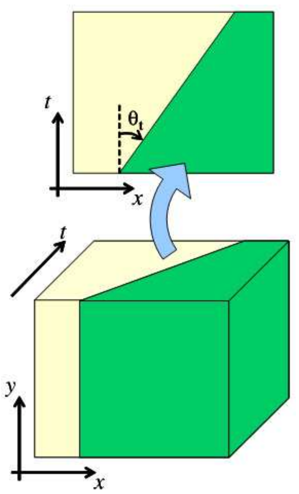
$$c_i \approx \alpha_i c_{i,s_i^1} + (1 - \alpha_i) c_{i,s_i^2}$$

where s_i^1, s_i^2 denotes the two segments, i is the pixel index, c_i is the colour contribution of pixel i, and c_{i,s_i^x} is the colour contribution of pixel i from segment s_i^x.

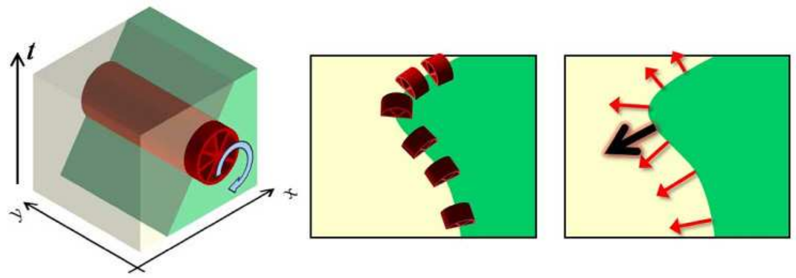
Another approach using layers to explicitly solve occlusions is given in Kumar et al. [12]. Overall, the technique is an unsupervised approach to generative layered representation for motion segmentation. It learns each rigidly moving object in a sequence and uses it in a layer. Once the parameters of the model have been estimated, any one of the frames in the sequence can be generated by selecting the right *latent variables*: represented by transformation, appearance, layer ordering and lighting parameters. An initial estimate of the model parameters is made using patches in an MRF framework, which segments motion using a loopy belief propagation technique. Given these initial model estimates, one of the methods it uses to improve the shape of the segments is α expansion. It expands each segment and checks for overlaps to see if either layer A occludes layer B or vice versa - by checking which configuration minimises the energy of the layered representation.

Ogale et al. [19] introduces an interesting formulation where it uses occlusions themselves for motion segmentation. It highlights 3 categories of object motion to differentiate motion due to camera and scene elements. The ordinal depth is computed according to the categorisation of the motion. If an optical flow estimate is provided, the method suggests a simple scheme to assign occluded regions to the right layer. 3 frames F_1, F_2, F_3 are given, with their optical flows u_{12} and u_{23}, and their reverse optical flow u_{21} and u_{32}. From previous computations, the method knows regions of occlusion O_{12} (regions present in F_1 but not in F_2), and O_{23}. We first consider finding the region labels of occluded regions in F_3, O_{23}. Given that the regions occluded in F_3 would have been visible in F_1, F_2; the method can segment the flow of u_{21} to find the labels for regions that subsequently got hidden in F_3 i.e. O_{23}. This helps in finding ordinal depth of the occluded regions.

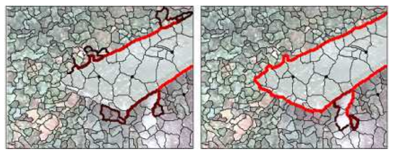
Xiao and Shah [36] assumes planar regions and extracts a set of affine or projective transformations that these regions undergo. In order to do this, it detects the occlusion pixels and segments the scene into motion layers. Using a level set formulation and *graph cut*s it creates an initial set of segments. In a two step process, it merges similar segments into layers on the basis of motion similarity. In the first step, two regions R_1, R_2 from the initial layers are merged if the SSD of the two regions, after applying the transformation H_2 on R_1, results in a majority of the pixels in R_1 supporting R_2. The motion parameters are recomputed after the merger. In the second step, the bi partitioning *graph cuts*



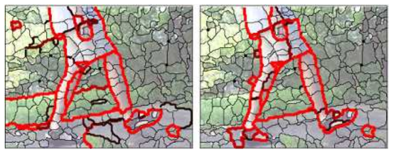
(a) A moving edge sweeps out an oriented path θ_t in a temporal slice



(b) A representation of the spatio-temporal edge detector used (left), shown at each point along a boundary fragment (mid), each detector will yield a combined normal (black) vector (right)



(c) Detection of occlusion boundary shown in red. Output generated with appearance (left) and motion+appearance (right) cues. The gray value of the edges denote the strenght of the edge



(d) Same as 2.2c shown on a different input

Figure 2.2: Shows the steps involved in resolved occlusion boundaries in Stein and Hebert [31]. Taken from Stein and Hebert [31]

algorithm is used to prune the un-supporting pixels from this new merged $R_1|R_2$ region. This results in layers, said to have coherent motion. Finally, it finds occlusion between overlapping layers using the graph cuts algorithm. This is done while ensuring consistency of layer segmentation using occlusion order constraints.

Stein and Hebert [31] also models motion discontinuities as occlusion events. After segmentation (described in Section 2.1.1) and motion estimation, it attempts to identify occlusion boundaries (which would belong to multiple surfaces). To detect such a moving motion edge, it considers the movement of the edges in a spatio-temporal volume (see Figure 2.2a). Since the tangent of the angle of this path (temporally) would correspond to the orientation and speed of edge's motion, it applies an oriented edge detector to the temporal slice of this volume. Here the choice of the filter is important. It uses a cylindrical detector capable of dividing data into two halves aimed at detecting significantly different distributions of motion (see Martin et al. [16]). Using this filter, it combines local normal speed estimates along the edge to get full 2D (u, v) motion estimate of the whole edge fragment (see Figure 2.2b). It uses a linear system of equations based on fragment's overall motion vector projected onto the local normal vectors:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \arg\min \sum_{i \in F} w(i)(n_{x,i}u + n_{y,i}v - s_i)^2$$

where $n_{x,i}$ and $n_{y,i}$ are the components of the normal at point i on the edge fragment F and s_i is the corresponding speed from the spatio-temporal detector. w_i denotes the contribution according to the local edge strength. These detected edge segments are critical to this technique, as they are associated with a set of motion-based and appearance-based cues. These cues are critical in computing the likelihood that an edge fragment is an occluding contour (see Figures 2.2c and 2.2d).

2.4 Algorithm suitability (Learned algorithm selection)

This section discusses a few works in algorithm selection as a classification task. The first two papers are generic machine learning papers, while the papers in the later half are related to algorithm selection in the domain of image segmentation, tracking and optical flow.

Muja and Lowe [18] concentrates on a common problem in many computer vision algorithms: nearest neighbour matching in a high-dimensional space. More formally, given a set of points $P = \{p_1, \cdots, p_n\}$ in a vector space X, and a query point $q \in X$, it tries to find the approximate nearest neighbour for q in P. To tackle this, the paper introduces a new method of approximate search which is a modification of the *hierarchical k-means tree*. But, in this section we are more interested with the algorithm selection technique it proposes. Given a dataset, this technique uses a cross-validation scheme, to select the ideal algorithm and its parameters. The paper demonstrates using this technique to select either *randomized kd-tree* or *hierarchical k-means tree* and suggests a set of parameters e.g. the number of iterations to use in case of the latter. The choice of algorithm is made by evaluating a cost of each algorithm, computed on a small subset of the input data:

$$\text{cost} = \frac{s + w_b b}{(s + w_b b)_{\text{opt}}} + w_m m_t / m_d$$

where s represents the search time, b represents the tree build-time, and m_t is the memory used for the tree, and m_d is the memory to store data. w_b is the weightage provided by the user for build-time i.e. when $w_b = 0$, the user isn't concerned with the build time. Similarly, w_m gives the importance of memory overhead against time-overhead. The time overhead is computed relative to $(s + w_b b)_{\text{opt}}$, which is the optimal search and build time if memory usage is not a consideration. Since the cost is built only to deal with nearest neighbour searches, adapting it to another method would require changing the cost altogether.

As a more general scheme, Raykar et al. [22] describes the problem of selecting the right label for a classification task given a number of algorithms (where the labels from algorithms might vary by a substantial amount) and no gold standard is given (for the classification problem). In order to find the true label, it presents a MAP estimator, which also learns the classifier, and the algorithm accuracies in parallel. The performance of each algorithm is judged on sensitivity and specificity with respect to the gold standard. Using these performance rating, each algorithm is ranked and assigned weightage that will be useful in all subsequent classification tasks. The EM algorithm is used to iteratively establish the final gold standard, using the performance measures from all the algorithms. The paper also mentions the possibility of using a beta prior (on sensitivity and specificity), in-case some prior knowledge about a particular algorithm needs to be asserted. This formulation is an improvement over a majority vote technique, which fails when some algorithms are more trustworthy than others.

An approach closer to our application is given in Yong et al. [37], which proposes to find the ideal segmentation algorithm for a particular image. The system can be categorised as a "performance prediction based algorithm selection model". This performance prediction is based on both image features, and prior knowledge and experience. Two assumptions are made to make this technique work: (1) the algorithm ideal for segmentation depends on image characteristics such as contrast, noise, and illumination; (2) the performance of an algorithm on images with similar characteristics is approximately the same. The method constitutes of three modules: (1) *performance predictor* which defines a prediction function based on segmentation quality of different algorithms, at the training stage; (2) *performance evaluator* is used in the training stage to rank the algorithms - which is a supervised offline scheme; (3) *feature extractor* which uses a simple intensity histogram. The ranks and the features extracted are eventually used to train the *performance predictor*. At runtime, the *performance predictor* ranks the segmentation algorithms according to the input image's (extracted) features. The paper demonstrates selecting the best algorithm in 85% of the test cases.

Moving from domain of segmentation, Stenger et al. [32] defines a similarly structured technique for tracking. Since tracking is an iterative process, it needs to learn the model of the to-be-tracked object, in hope of detecting it in subsequent frames. Here, an issue that can plague a learning technique in tracking, and any other iterative classification approach (unlike segmentation), is that of adaptability against drift. The tracker can learn (adapt) the object model quickly but it will be more susceptible to drifting off (by erroneously learning the background model). To avoid this scenario, [32] suggests using a trained offline detection component. Each tracker is also evaluated during this training stage. Every tracker O^k outputs the estimate of the tracked position x_t^k on a training set, its error e_t^k from the ground-truth, and a confidence value c_t^k. Selecting a particular threshold on error, the loss of track can be identified automatically. Using these measures, *precision* ($1-$ expected error) and *robustness* (the probability of keeping track) for each tracker is noted. The method proposes two schemes for online tracking. The first one is a *parallel* scheme, where multiple trackers are used simultaneously. The tracker with the lowest expected error given the confidence value ($\min_k E[e^k|c^k]$) is used for the output. The second is a *cascaded* scheme, where trackers are used sequentially until a tracker is found whose expected error is below a threshold ($E[e^k|c^k] > \tau \rightarrow$ evaluate next tracker). The advantage of using a sequential approach is the reduced computational time.

Mac Aodha et al. [14] suggests a novel approach to evaluating optical flow by choosing ideal algorithms per-pixel. It supposes that some form of gold standard is available during the training phase but not during runtime. Like Yong et al. [37], it also tries to find a relation between good performance of an algorithm against specific spatial and temporal features. Experimental results given, use Random forests (see Breiman [4]) to classify which flow algorithm (4 algorithms considered - $k = 4$) to use at a particular pixel. A total of 44 features were input to the classifier which included measure of textured-ness, distance from edges, derivative of proposed flow field (to be wary of motion discontinuities), all at different image pyramid levels. The best features were selected by the classifier and used in the algorithm selection process.

As an added note, if we consider interest point detectors, like Shi and Tomasi [29], as binary classifiers on image features, they already provide a rudimentary base for algorithm selection. Although an advantage of these schemes is that no energy functional needs to be employed over the complete data to get optimal results.

Chapter 3

Approach to Segmentation and Optical Flow

The aim of this thesis is to explore techniques in both segmentation fused with flow, and algorithm suitability techniques. The first part of the thesis would aim to develop a novel iterative approach for segmentation and flow. The second part of the thesis we will use the developed iterative algorithm as a regularisation constraint in an algorithm suitability approach.

3.1 Iterative scheme for image Segmentation and Optical Flow

The aim of this approach would be to explore techniques that are relevant segmentation, optical flow (tracking) and motion segmentation. Rather than trying to develop segmentation and optical flow schemes from scratch, the work will aim towards using of-the-shelf techniques and combining them in a useful configuration.

Here we propose using an optical flow technique which uses a graph-based formulation. Such techniques are quite flexible and allow the use of higher-level framework for not only regularisation but also segmentation. Shi and Malik [27] gives a great example of such an approach - which segments the graph using normalised cuts. Also, it defines *motion profiles* useful for comparing segment motions. A similar approach is suggested in Galun et al. [7] which uses *motion profiles* without building a graph.

For segmentation, a graph based *superpixel* method would be recommended, since it will allow for easy fusion with a graph based approach used for optical flow. An over-segmentation technique is suggested, because Wang and Adelson [34] shows that simpler motion models are not accurate representations of large segments. Here, we Yu and Shi [38] provides a good starting point as it develops a pixel graphs and *edgel* graph for segmentation. It also has the advantage of using both region and boundary information. The problem with using this approach is that it doesn't aim to over-segment the image. A good alternative would be to use [6] which treats each node as a pixel in the graph and produces over-segmentations. Although it has two shortfalls: (1) it does not use a compactness constraint; (2) its segments are shown to be a degenerate case of Levinshtein et al. [13].

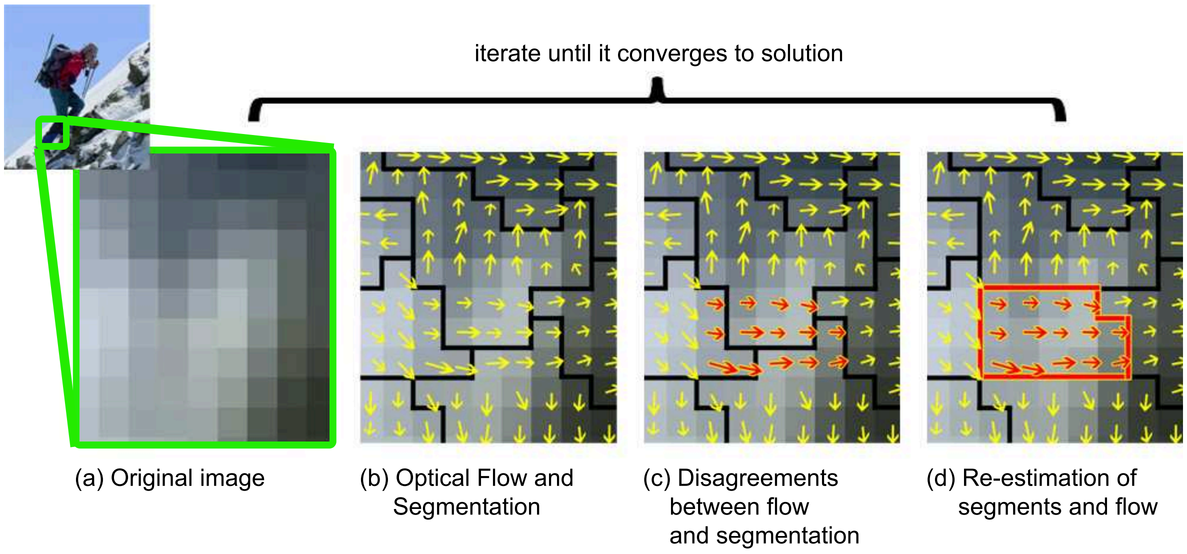

iterate until it converges to solution
(a) Original image
(b) Optical Flow and Segmentation
(c) Disagreements between flow and segmentation
(d) Re-estimation of segments and flow

Figure 3.1: Displays the steps (for a single iteration) of the proposed approach for iterative image segmentation and flow computation

It might be possible to skip a graph-based approach, are replace it with a geodesic contour evolution algorithm like Levinshtein et al. [13].

Figure 3.1 the basic steps that are proposed for this thesis' image segmentation and optical flow estimation approach. Since segmentation and optical flow are related problems, as mentioned in Section 1.4, an iterative approach might be able to improve the results of segmentation and optical flow. This approach is similar to Zitnick et al. [45, 44] - although this thesis doesn't plan to take a generative approach.

The main steps of the algorithm would be: (1) segmentation Levinshtein et al. [13] and optical flow [27] computation; (2) Noting disagreements between the flow and segmentation (could be done by an k-way *graph-cuts* implementation); (3) Re-etimation of the flow field and the segments. Since this would be an iterative process, a stopping condition would need to be defined. Once this algorithm is developed, the thesis will concentrate on its application in algorithm suitability.

3.2 Algorithm suitability

The techniques discussed in Section 2.4 give a key insight in the choice of approaches that could be used. Both Yong et al. [37] and Mac Aodha et al. [14] indicate the feasibility of an offline training phase followed by a runtime with no given gold standard. One option could be to use a fusion of these two approaches to decide on an ideal segmentation of flow in the image plane.

Although a developing a regularisation scheme would be of greater interest in this thesis. A critical component missing in Mac Aodha et al. [14] is a regularisation constraint on the choice of algorithms in a neighbourhood. Each pixel in the method is computed for suitability for all different techniques, which is not only a computationally intensive task, but also might be un-necessary. Given the algorithm developed in the previous section, we might be able to give accurate estimates of the motion segments in a pair of images. Deciding for a single pixel's algorithm suitability in each segment might work well as a regularisation constraint. Of course, this supposes that a tracking technique best for a pixel is also ideal for the rest of the pixel's homogenous segment. Also since we will have an initial flow estimated, this can be used as one of the k algorithms selected by [14].

Chapter 4

Conclusion

We mainly explored multiple techniques for segmentation, motion analysis, and motion segmentation - some of them might be useful in fusion based approaches to compute eventual segmentations/flow fields. We also explored algorithm suitability techniques and how they can be defined as a classification problem. Their applications were discussed, including in tracking and segmentation.

Section 3 gave a brief overview of the possible approaches that will be developed during the progress of this thesis. We discussed how a graph based technique can be combined in an iterative approach with superpixel segmentation. We also proposed how this method can be applied to an algorithm suitability approach as a regularisation constraint.

Bibliography

[1] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Computer Vision ECCV '92*, pages 237–252. Springer, 1992. 16

[2] M. Black. Recursive non-linear estimation of discontinuous flow fields. *Computer Vision ECCV '94*, pages 138–145, 1994. 13

[3] MJ Black and AD Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):972–986, 1996. 13

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 20

[5] M.M. Chang, A.M. Tekalp, and M.I. Sezan. Simultaneous motion estimation and segmentation. *IEEE Transactions on Image Processing*, 6(9):1326 –1333, sep 1997. 12, 14

[6] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004. 8, 10, 21

[7] M. Galun, A. Apartsin, and R. Basri. Multiscale segmentation by combining motion and intensity cues. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '05)*, volume 1, pages 256–263, june 2005. 13, 15, 21

[8] F. Heitz and P. Bouthemy. Multimodal estimation of discontinuous optical flow using Markovrandom fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12):1217–1232, 1993. 11

[9] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 12

[10] M. Irani and P. Anandan. Factorization with uncertainty. *Computer Vision ECCV '00*, pages 539–553, 2000. 16

[11] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988. 9

[12] M.P. Kumar, P.H.S. Torr, and A. Zisserman. Learning layered motion segmentations of video. *International Journal of Computer Vision*, 76(3):301–319, 2008. 17

[13] A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, and K. Siddiqi. TurboPixels: Fast Superpixels Using Geometric Flows. *IEEE transactions on pattern analysis and machine intelligence*, pages 2290–2297, 2009. 10, 11, 21, 22

[14] O. Mac Aodha, G.J. Brostow, and M. Pollefeys. Segmenting Video Into Classes of Algorithm-Suitability. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '10)*, 2010. 20, 22, 23

[15] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001. 3, 4

[16] D.R. Martin, C.C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004. 10, 18

[17] R. Megret and D. DeMenthon. A Survey of Spatio-Temporal Grouping Techniques. Technical Report CS-TR-4403, Language and Media Processing, University of Maryland, College Park, MD, August 2002. 16

[18] M. Muja and D.G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISSAPP '09)*, volume 331–340, 2009. 19

[19] A.S. Ogale, C. Fermuller, and Y. Aloimonos. Motion segmentation using occlusions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):988–992, 2005. 17

[20] N. Paragios and R. Deriche. Geodesic active regions for motion estimation and tracking. In *Proceedings of the Seventh IEEE International Conference on Computer Vision, (ICCV '99)*, volume 1, pages 688–694, 1999. 14

[21] N. Paragios and R. Deriche. Geodesic Active Regions: A New Framework to Deal with Frame Partition Problems in Computer Vision. *Journal of Visual Communication and Image Representation*, 13(1-2):249–268, 2002. 9, 14

[22] V.C. Raykar, S. Yu, L.H. Zhao, A. Jerebko, C. Florin, G.H. Valadez, L. Bogoni, and L. Moy. Supervised Learning from Multiple Experts: Whom to trust when everyone lies a bit. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 889–896. ACM, 2009. 19

[23] X. Ren and J. Malik. Learning a classification model for segmentation. In *Proceedings of the Ninth IEEE International Conference on Computer Vision, (ICCV '03)*, volume 1, pages 10–17, oct. 2003. 10

[24] M.G. Ross. Exploiting Texture-Motion Duality in Optical Flow and Image Segmentation. Master's thesis, Massachusetts Institute of Technology, Aug 2000. 4, 9

[25] P. Schroeter and J. Bign. Hierarchical image segmentation by multi-dimensional clustering and orientation-adaptive boundary refinement. *Pattern Recognition*, 28(5):695–709, 1995. 9

[26] E. Sharon, A. Brandt, and R. Basri. Segmentation and boundary detection using multiscale intensity measurements. In *Proceedings of the 2001 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '01)*, volume 1, pages 469–476, 2001. 15

[27] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *Proceedings of the Sixth IEEE International Conference on Computer Vision, (ICCV '98)*, pages 1154–1160, jan 1998. 5, 12, 13, 14, 21, 22

[28] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. 9, 10

[29] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the 1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '94)*, pages 593–600, 1994. 20

[30] C. Stauffer and W.E.L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000. 15

[31] A.N. Stein and M. Hebert. Occlusion Boundaries from Motion: Low-Level Detection and Mid-Level Reasoning. *International Journal of Computer Vision*, 82(3):325–357, 2009. 10, 18

[32] B. Stenger, T. Woodley, and R. Cipolla. Learning to track with multiple observers. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pages 2647 –2654, june 2009. 20

[33] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based onimmersion simulations. *IEEE transactions on pattern analysis and machine intelligence*, 13(6):583–598, 1991. 10

[34] J.Y.A. Wang and E.H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, Sept. 1994. 16, 21

[35] K.Y. Wong and M.E. Spetsakis. Motion segmentation and tracking. In *Proceedings of 15th International Conference on Vision Interface*, pages 80–87, 2002. 13, 15

[36] J. Xiao and M. Shah. Motion layer extraction in the presence of occlusion using graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1644–1659, 2005. 17

[37] X. Yong, D. Feng, Z. Rongchun, and M. Petrou. Learning-based algorithm selection for image segmentation. *Pattern Recognition Letters*, 26(8):1059–1068, 2005. 19, 20, 22

[38] S.X. Yu and J. Shi. Perceiving Shapes through Region and Boundary Interaction. Technical Report CMU-RI-TR-01-21, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2001. 9, 21

[39] S.X. Yu and J. Shi. Multiclass spectral clustering. In *Proceedings of the Ninth IEEE International Conference on Computer Vision, (ICCV '03)*, volume 1, pages 313–319, oct. 2003. 10

[40] S.X. Yu, R. Gross, and J. Shi. Concurrent Object Recognition and Segmentation by Graph Partitioning. In *Advances in neural information processing systems 15*, pages 1383–1390. MIT Press, 2002. 9

[41] L. Zelnik-Manor, M. Machline, and M. Irani. Multi-body factorization with uncertainty: Revisiting motion consistency. *International Journal of Computer Vision*, 68(1):27–41, 2006. 16

[42] C.L. Zitnick and S.B. Kang. Stereo for image-based rendering using image over-segmentation. *International Journal of Computer Vision*, 75(1):49–65, 2007. 11

[43] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. *ACM Transactions on Graphics*, 23(3):600–608, 2004. 11, 17

[44] C.L. Zitnick, S.B. Kang, and N. Jojic. Simultaneous optical flow estimation and image segmentation. US Patent 7,522,749, Apr 2009. Filed: July 2005. 17, 22

[45] C.W. Zitnick, N. Jojic, and Sing Bing Kang. Consistent segmentation for optical flow estimation. In *Proceedings of the Tenth IEEE International Conference on Computer Vision, (ICCV '05)*, volume 2, pages 1308–1315, oct 2005. 9, 17, 22