Multiple Object Tracking with Occlusion Handling

by

Murtaza Safri

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Object tracking is an important problem with wide ranging applications. The purpose is to detect object contours and track their motion in a video. Issues of concern are to be able to map objects correctly between two frames, and to be able to track through occlusion. This thesis discusses a novel framework for the purpose of object tracking which is inspired from image registration and segmentation models. Occlusion of objects is also detected and handled in this framework in an appropriate manner.

The main idea of our tracking framework is to reconstruct the sequence of images in the video. The process involves deforming all the objects in a given image frame, called the initial frame. Regularization terms are used to govern the deformation of the shape of the objects. We use elastic and viscous fluid model as the regularizer. The reconstructed frame is formed by combining the deformed objects with respect to the depth ordering. The correct reconstruction is selected by parameters that minimize the difference between the reconstruction and the consecutive frame, called the target frame. These parameters provide the required tracking information, such as the contour of the objects in the target frame including the occluded regions. The regularization term restricts the deformation of the object shape in the occluded region and thus gives an estimate of the object shape in this region. The other idea is to use a segmentation model as a measure in place of the frame difference measure. This is separate from image segmentation procedure, since we use the segmentation model in a tracking framework to capture object deformation. Numerical examples are presented to demonstrate tracking in simple and complex scenes, alongwith occlusion handling capability of our model. Segmentation measure is shown to be more robust with regard to accumulation of tracking error.

Acknowledgements

Firstly and formostly, I wish to express my deepest gratitude for my supervisor and mentor Justin Wan, without whose guidance this piece of research would not be possible. His breadth of knowledge in different areas of mathematics and computing was immensely helpful throughout the course of my work. His emphasis on rigourous thinking and questioning of established dogma leading to new vistas for scientific exploration, will continue to inspire me in my future pursuits.

I am deeply appreciative of my thesis committee members, Yuying Li and Jeffery Orchard for their time and advice towards the completion of this thesis.

Special thanks to my friends and colleagues, David Zahedi, Tian Tian Bian, Stephen Tse, Laura Bradbury, Dong Han, and Bo Wang, for their help and participation in recording the experimental video clips used in this thesis.

Lastly, but not the least, I would like to thank my family and friends for their love and care. I am indebted to my parents for their staunch support and encouragement. I would also like to thank all of my colleagues in Scientific Computing lab for providing a friendly and intellectually healthy atmosphere.

Dedication

This is dedicated to my parents, Shabbir Safri and Jahanara Safri.

Contents

List of Figures

Chapter 1

Introduction

In the field of computer vision there are various aspects of interest such as the abilitiy to detect and represent objects present in an image, to observe motion of objects in video frames and to derive a cognitive interpretation of the motion for understanding object's behavior. Tracking of objects is an important open problem in computer vision. It has widespread applications, for instance, in robotic navigation to traverse a path while avoiding obstacles, security surveillance for monitoring individuals for suspect behavior [32], monitoring vehicular movement at highways and traffic intersections, astronomy for observing the motion of stellar bodies relative to earth, medical imaging for tracking the motion of organs [25], such as the heart [56], brain, and blood vessels, and, life sciences for tracking moving cells in a culture. One particular example of interest is tracking of cells in brightfield microscopy [46]. This problem is difficult because the image constrast is low, and the cells appear similar to each other and the background. Also, certain types of cells, such as muscle cells, change shape rapidly and can occlude each other in complex ways.

The main objective for tracking is to identify objects in a sequence of frames of a video and determine where each object moves in the scene. Several approaches with which this can be done are tracking certain feature points of the object [41], tracking a kernel over the object region [7], or contour tracking of the objects [3]. There are several ways of detecting objects in scene, depending on the type of tracking method used. For point tracking, feature points of the object have to be extracted. SIFT, KLT, and Harris corner detector are examples of methods commonly used for feature point extraction. Region and contour tracking require the region of interest is first

detected using segmentation methods.

Tracking of objects is challenging for several reasons. One may need to deal with complexities such as noise in images, varying illumination with time, loss of information in two dimensional (2D) images of three dimensional (3D) scenes, and complex object motion. Multiple object tracking poses its own set of problems. For instance, if the objects resemble each other in appearance, then although objects may be detected it becomes difficult to distinguish between themselves. It is necessary to tell objects apart from each other so that correspondence between detected objects in two different frames may be established, and, consequently object motion be obtained by comparing past positions with current ones. Another problem is dealing with occlusion which occurs when part or all of the object is hidden from the vantage of observation. For example, objects moving in a video may overlap and hide other objects. As a result, information about the occluded part is lost. In practice, however, it is often desirable to estimate the object shape in the occluded region. Consider a case where an object moves underneath, say, from one side of a square object. Without the occluded shape information, when the object reappears from the other side of the square, it would be difficult to tell whether it is the same object or a new one. Shape information helps to match occluded part of the object with the same object in the next frame. The method for recovering the lost information is to impose some constraints on the behavior of the objects. Objects are normally assumed to be undergoing "predictable" motion in the image sequence such that there can be no significant changes in the shape, features, appearance, velocity or acceleration of the object from frame to frame.

Our approach to tracking is to compute the motion of objects. Our model is based on image registration model which is used to compute the displacement. More precisely, given object and background information in a frame, we reconstruct the next frame using the idea of registration. The displacement field over the whole object is then computed and it is used to establish correspondence between objects in consecutive frames. It can also be used to recover the shape of an object in the occluded regions as well. The registration framework is used for computing the displacement. Most approaches resort to finding the segmentation contour for tracking the object whereas we find the displacement field of the object contour. One benefit of this approach is that it will give the contour of the occluded region implicitly without having to detect occlusion explicitly. Another benefit is that our model is computationally less expensive than the typical segmentation based approaches, since the number of

model unknowns is often less in our case. To measure the quality of reconstruction, we use L^2 norm of the difference of reconstructed image and the actual image in the video. Our model uses level sets for representing objects. This allows us to seamlessly integrate any existing segmentation model based on the level set framework as a similarity measure in our tracker. We demonstrate this idea with a segmentation measure and compare its robustness with image difference measure. When segmentation measure is used in a registration framework, it is distinct from segmentation algorithms. In the former, the unknown is object motion, whereas in the latter, the unknown is the segmentation contour.

The rest of this thesis is organized as follows. We describe the various approaches for object tracking in Chapter 2. The focus in this chapter is on discussing existing contour tracking models and their relation to our model. In Chapter 3, we provide a brief review of the image registration and segmentation techniques. A particular model for each one is described, which will later be used in our tracking model. In Chapter 4, we present our tracking model which is based on the rigid and nonrigid registration. We also discuss how we combine the rigid and nonrigid registration based tracking to handle occlusions. The segmentation model is introduced into the registration based tracking model as a similarity measure. The segmentation based similarity is then compared with the image difference based similarity. Later, we derive the Euler-Lagrange equations for minimizing the tracking model. We also make certain modifications to the model so that it is numerically accurate to solve. In Chapter 5, we discuss the implementation details for solving the rigid and nonrigid steps. Levenberg-Marquardt method is used for the computing the rigid step and finite difference method for the nonrigid step. Computational performance issues are also encountered and remedies are devised to reduce the computational complexity. Finally, tracking examples are given in Chapter 6 to demonstrate our method.

Chapter 2

Background Review

Computer vision tracking has been studied extensively in the literature. The are two main steps in object tracking, one is to detect moving objects in an image, and other is to track objects in sequence of images. A survey by Yilmaz and Shah [54] gives a thorough review and classification of techniques into the following major categories - point tracking, kernel tracking and silhoutte tracking. These categories are based on the representation of objects used for tracking. In this chapter, we will discuss some of the methods that fall into these categories. The review of layers based methods, which fall into both kernel and silhoutte tracking, is conducted at the end.

2.1 Point Tracking

In point tracking methods, the object is represented by a single point or set of points. In the case of a single point, it is usually the centroid of the object. Single points are suitable for representing small objects in an image such as birds in a flock or cells in microscopic images. If the object is relatively large then multiple points on the object are used for representation. These points represent some local features in a region of the object such as a corner, intensity gradients or a particular color and texture. Examples of point detectors are KLT [44] and SIFT [27] transforms. The goal in point tracking is to detect points of interest in video frames and to establish correspondence between points in the frames. Mainly, there are two classes of correspondence methods, deterministic and statistical.

In deterministic methods, each correspondence between two points has an associated cost. By minimizing the total cost for all points correspondence is established. The costs are based on constraints imposed on the motion from one frame to the other. It is assumed that object does not move rapidly from frame to frame so that the displacement is bounded within a window around the object. That the object moves smoothly without large changes in speed and direction, is also a frequently used assumption. There may also be constraints on the shape of the object, i.e., it moves rigidly so that the relative position of adjacent points remains the same. One way is to use nearest neighbor search for correspondence. The approach by Veenman et al. [47] imposes motion constraint that close points have similar motion.

In statistical methods, the position, velocity, and acceleration of the object is modeled in the state space. Uncertainities and noise in the state space model, such as abrupt change in the velocity of objects, are also taken into account. The state of the system at a given time is computed from the state at a previous time instants. For single objects Kalman filters [10] are used for predicting the state if the distribution of the state random variables is Gaussian. Particle filters [17] are used when the type of distribution is unknown. For multiple objects, Joint Probability Data Association Filtering (JPDAF) [36] and Multiple Hypothesis Tracking (MHT) [38] are used. MHT exhaustively generates all possible associations over several frames and then selects one which has the highest likelihood.

Point tracking is generally not suitable for highly deformable or nonrigid object motion, since in general, the shape of an object cannot be related to the position of its features points. For the same reason, it is not suited for recovering occluded regions. Also, points may not be recovered correctly once they become occluded if the motion constraint assumptions do not hold. When multiple large objects are being tracked, with several points on each object, there is the additional complexity of associating the points correctly to their respective objects which are in vicinity of each other.

2.2 Kernel Tracking

In this category of tracking methods, the object is represented by a kernel which is a primitive geometric shape, such as rectangle or ellipse, superimposed upon the object. The centroid of the kernel is considered to be the location of the object. The object's

motion is modeled by rigid, affine, or projective transformation which is applied to the kernel for the updated location. The position of the kernel is determined by matching the appearance characterisitics of the object within an image region.

Template matching may be used to locate the object in the target frame. The position of the template is computed by minimizing a similarity measure, such as the sum of squared differences or maximizing the correlation coefficient. Image intensity is normally used as the feature in the template images. However, other features such as texture or edges may also be used.

Statistical models of appearance are commonly used in kernel tracking. Comanicius et al. [14] model the kernel with a weighted histogram, with weight inversely proportional to the distance of the pixel from the kernel center. The motion is given by the mean shift tracker by computing similarity of the histogram of the object and the histogram of the window around the hypothesized target location. The distance measure used to compute the similarity is the Bhattacharya coefficient [6]. They also extend the tracker by incorporating Kalman filter for improved search of the target. Optical flow is also used for computing the location of the kernel by computing the displacement field that enforces the brightness constraint. The KLT tracker [44] proposed by Shi and Tomasi computes the motion of a rectangular region by computing the optical flow of the feature points detected by the tracker.

Khan and Shah [23] developed an algorithm for tracking people under occlusion. They do it by tracking the person as separate regions connected together. However, in their approach there is no recovery of occluded regions. Vezzani et al. [15] track an object as a macro object composed of smaller regions called tracks. Once regions have been detected based on appearance, they are joined into a macro object. Again in this approach, there is no recovery of occluded region as it is a kernel based method.

Isard and MacCormick [16] track multiple objects by joint modeling of the background and foreground regions with Gaussian distributions. The shape of objects are modeled as cylinders. Particle filters are used for predicting the state vector which includes the velocity, position and shape. Their method is also capable of tolerating partial occlusions.

Representing objects with simple geometric shapes has inherent problems. Certain parts of the objects may lie outside the kernel region while parts of the background may fall within it. This makes it difficult to correctly classify pixels inside the kernel. Kernel tracking is works suitably for objects that appear largely rigid. For nonrigid

objects, kernel tracking may work but it is possible that object regions fall outside the kernel, leading to errors.

2.3 Silhoutte Tracking

In silhoutte trackers, the object is represented by a contour including the region inside the contour. Tracking can be either boundary based or region based. For boundary based approach, the contour is evolved to match the boundary edges of the object. In region based approach, the contour is evolved until the region inside the contour has the same appearance as the object. The appearance of the object can be for example the intensity image of the object. It can also be a statistical measure such as mean intensity and standard deviation of the object region. Shape of the object is another type of information that can be used along with apperance. Object information could either be selected from the object region in the previous frame or it could be statistically learnt information over the past few frames. The region based approach and kernel tracking both use the appearance information of the object.

Since their introduction by Kass et al. [21], active contour have been used extensively as a deformable model for object detection [21, 8, 26]. Active contour models allow detetction for object of any shape by detecting strong gradients around the boundary. The initial contour is assumed to enclose the object of interest and shrunk so that it locks onto the object boundary. Casselles et al. [12] reformulated the curve evolution problem as finding the geodesic (minimal distance curve) in Riemannian space for detecting boundaries. The geometric flow approach improves the detection capability of active contours by preventing the contour from seeping in through small holes in the boundary. Paragios and Deriche [34] use the interframe difference histogram as a Gaussian distribution to classify pixels as belonging to the boundary of a mobile or static object. They set this up in the geometric flow framework and so build upon object detection technique from Caselles [12].

Edge detection based active contours do not perform well on complex and textured backgrounds. To overcome, this Ronfard [40] and Zhu and Yuille [57] proposed a region based energy where object and background regions are statistically modeled. The contour evolves outwards if the neigborhood matches object model and inwards if it matches background.

Nonrigid motion can be computed as optical flow given by the brightness constancy constraint. Bertalmio et al. [3] computes optical flow in a band around the boundary by solving two PDEs, one for evolving the contour and the other for intensity morphing. This method is useful only when object deformation is small. Mansouri [30] uses a probabilistic brightness constraint modeled by a Gaussian distribution. He performed a search for the optical flow vectors for each pixel in a circular neighborhood.

Yilmaz and Shah [52, 55] developed a general framework for tracking that is based on previous work in region and boundary based tracking. Their model learns shape and color online and applies it to obtain occluded regions. We do not adopt learning techniques in our model as it may not be suitable for tracking highly deformable objects, which have similar shapes for several frames but later change shape abruptly. However, we remark that learning based model can be incorporated in our framework if the application area can take benefit from this additional information. Also, in order to detect occlusion, Yilmaz and Shah [55] use a heuristic method. When the object becomes occluded, there is a significant reduction in the area of the object as compared to the average area in the previous frames. If the ratio of current area to the average area is below a user specified threshold parameter, the object is treated as occluded. Once occlusion is detected the contour in the occluded region is recovered using shape level sets. This heuristic may not work well for highly deformable objects, as it is not clear how to distinguish whether reduction in area is due to occlusion or actual change in the size of the object.

Yezzi and Soatta [53], and Jackson et al. [18] developed a model to describe the average shape of a deforming object being tracked. The object motion is decomposed as a global rigid or affine motion, and a local deformation. This is similar to how we treat object motion. However, the difference is that their approach assumes object segmentation is available. They predict shape by learning the shape average over several frames. Their model also has no explicit occlusion detection, so if there is an occlusion lasting several frames a wrong shape with much smaller area would be learnt. Another approach to predict shape is to use particle filters for infinite dimensional state space, [37]. They set the state variable to be the affine (global) transformation and a curve for local deformations. Their assumption for shape prediction is that the object undergoes constant acceleration, which may not necessarily hold, for example, if the object moves in a jagged manner or is deforming rapidly over each frame.

2.4 Layers based Tracking

For multi object tracking with occlusion handling, layers based representation of the scene is appropriate. Each object is represented by a layer which has the appearance and shape information. The background, which contains objects that are stationary, is represented by its own layer. Additionally, a depth order is assigned to each layer to describe the relative position of the objects in terms of distance from the observer. There are methods which use both kernel and silhoutte representation in a layers framework.

Wang and Adelson [49] are one of the earliest to use a layers based representation of the scene. They first compute the optical flow between two frames to obtain the global motion. The optical flow field has discontinuities at boundaries of moving objects. The field is clustered where it is smooth, in order to separate the motion of objects from each other. The localized optical flow is then fit to an affine motion model with linear least squares, to compute the motion parameters of each layer. The shape of the object is modeled with a binary image. Occluded regions are recovered by applying the median filter operation on the layer pixels. Depth ordering is established by counting the number of pixels in each layer compared with the motion compensated layer. In the occluded layers the number of pixels would be reduced compared to unoccluded layers.

Kernel representation of objects in layers is used by Tao et al. [45]. The kernel is an ellipse and the model parameters are the rigid motion and layer appearance, both of which are modeled by a Gaussian distribution. They compensate for the background motion by modeling it as projective transform. The farther points from the center in the kernel are assigned a probability of the background and the nearer points are assigned the probability of the object, in the layer. For each pixel in the image, a probability of its assignment to a particular layer is computed based on the past motion and the appearance probability. They compute the model parameters iteratively with the expectation maximization algorithm, by fixing layer ownership and estimating the motion, then fixing the motion and estimating the layer ownership.

Jackson et al. [19] developed a layers framework with deformable objects. The motion of the layers is composed of a group transform and local deformation applied on the intensity template. The transform parameters are selected by minimizing the difference between the topmost layer at a point in the domain and the video frame,

and the occlusion boundaries are recovered from the layers that are hidden by others. Their model is also capable of inpainting layers where the intensity information of a regions remains occluded.

Chapter 3

Image Registration and Segmentation

Image registration is the task of finding a transform between two images such that they are aligned in an optimal configuration. Motion based tracking requires registration of objects from frame to frame. The objective is to compute a transformation that maps an object from one frame to the next one. This idea is similar to image registration where the template image is mapped to the reference image, with the difference being the former operates individually on objects in an image while the later operates on whole images.

Segmentation is the process of finding the boundaries of objects in an image. It is often used for tracking by detecting objects in a video in consecutive frames and establishing correspondence by object appearance. This chapter discusses the key elements of image registration and segmentation model which will be used in later chapters for formulating our tracking model.

3.1 Images

In digital representation, images are a discrete set of values, called pixels, in an image domain. Each pixel is characterized by its location in the domain and its intensity value. The locations of the pixels are discrete values, and normally, the intensity

values are in the range $[0, 255]$ for an 8 bit encoded image. Mathematically, however, we treat an image as a continuous function

$$f : \Omega \to \rho$$

where $\Omega = \mathbb{R}^d$, $\rho = \mathbb{R}^q$, $d \in \mathbb{N}$ is the spatial dimension, and $q \in \mathbb{N}$ is the color dimension of the image. Examples are 2D images with $d = 2$, 3D images with $d = 3$, grayscale images with $q = 1$, and color images with $q = 3$. For simplicity the image domain and intensity values are scaled so that $\Omega = [0, 1]^d$ and $\rho = [0, 1]^q$. For most part of the thesis two dimensional grayscale images will be considered, while later, extensions will be made to two dimensional color images.

3.2 Image Registration

In this section, we give a brief review of image registration, its applications, and, classification. Later we provide the general mathematical formulation of registration. The similarity measures and spatial transformations used in the process are also described.

3.2.1 General Description

Image registration is used for bringing two or more images into alignment with each other so that the information in them may be combined. The images may have been taken at different times so that they are out of alignment due to motion of the capturing device or the scene. Images of the same scene may have been captured using different sensors and devices so that different types of information may be acquired. They may also be taken from a different pose of the same scene.

Image registration has many applications in different fields such as computer vision, medical imaging, remote sensing, astronomy etc. In medical imaging, for example, images taken from different devices such as CT and MRI show different features of the same anatomy. By overlaying the images from the two devices, the information for the same spatial region may be seen together. In geospatial imaging, satellite images taken at different times may be compared together for observing changes in the landscape. In computer vision, registration is relevant to the task of sterescopic vision, which is to

capture two dimensional images from different view points an use them to reconstruct a three dimensional scene.

Registration may be classified into different categories based upon the type of application [58]. One such classification is made on registration basis i.e. the features that drive the registration. Landmark and intensity based registration are the primary categories in this classification. In landmark based registration (see [39] for details), the images are first pre-processed to detect features such as points, edges, or shape. The images are then registered so that corresponding features in different images are overlayed together. Intensity based registration instead works by registering intensities directly in the images, [2, 20, 29]. In this thesis, we will be concerned only with intensity based registration. Another classification is based on the transformation types used for registering the images. Transformations fall into two broad categories - global transforms such as rigid, affine, and projective, and, local transforms. The details for the transformation types are dicussed below. One more classification is that of mono-modality registration in which the images are captured from the same sensors, and multi-modality registration, in which the images are from different sensors. In this thesis, we are concerned only with mono-modality images.

3.2.2 Mathematical Formulation

Formally, the purpose of image registration is to compute the spatial mapping of the template image $A : \Omega_A \rightarrow [0, 1]$ to the reference image $B : \Omega_B \rightarrow [0, 1]$, i.e.,

$$\Psi : \Omega_B \rightarrow \Omega_A,$$

so that the deformed template $A(\Psi)$ becomes similar to the reference image B. Registration is then stated as an optimization problem

$$\max_{\Psi} \mathcal{S}\left(A(\Psi), B\right),$$

where $\mathcal{S}$ is the similarity measure between the deformed template and the reference image. An alternative way to state registration is to minimize the distance between the transformed template and the reference image. Then image registration is stated as

$$\min_{\Psi} \mathcal{D}\left(A(\Psi), B\right),$$

where $\mathcal{D}$ is the distance measure. The objective function used in the optimization is called the similarity measure. In this thesis, we will simply refer to the distance measure, stated in the minimization problem, as the similarity or image difference measure.

3.2.3 Similarity Measure

For images captured by similar devices in such a way that there is a direct correspondence between the intensities in the two images, the image difference can be used as the distance measure. One distance measure commonly used is the Sum of the Squared Differences (SSD), which is the L^2 norm of the difference between the transformed image $A(\Psi(\mathbf{x}))$ and the reference image $B(\mathbf{x})$,

$$\mathcal{D}^{SSD}[\Psi] \equiv \int_{\Omega} \frac{1}{2} \left(A(\Psi(\mathbf{x})) - B(\mathbf{x})\right)^2 d\mathbf{x}. \tag{3.1}$$

This distance measure would be minimized once the images are in correct alignment. Another measure is the Sum of Absolute Differences (SAD), which uses the L^1 norm over the image differences,

$$\mathcal{D}^{SAD}[\Psi] \equiv \int_{\Omega} |A(\Psi(\mathbf{x})) - B(\mathbf{x})| d\mathbf{x}. \tag{3.2}$$

SAD has a benefit over SSD in that it is less sensitive to large variations in image intensities. However, the optimization with SSD is more convenient to solve numerically as compared to SAD; the former can be solved with Gauss-Newton or Levenberg Marquardt method whereas the latter cannot.

3.2.4 Transformation Types

The transformations used to align the template image are generally categorized as global and local transformations. The global transforms act on the whole image domain, while the local transform produce spatially localized deformations.

Rigid, affine, and projective fall within the class of global transformations. Rigid and affine transforms are shown in Figure 3.1. These are linear transformations that are represented by a linear operation on the spatial coordinates. The rigid transform

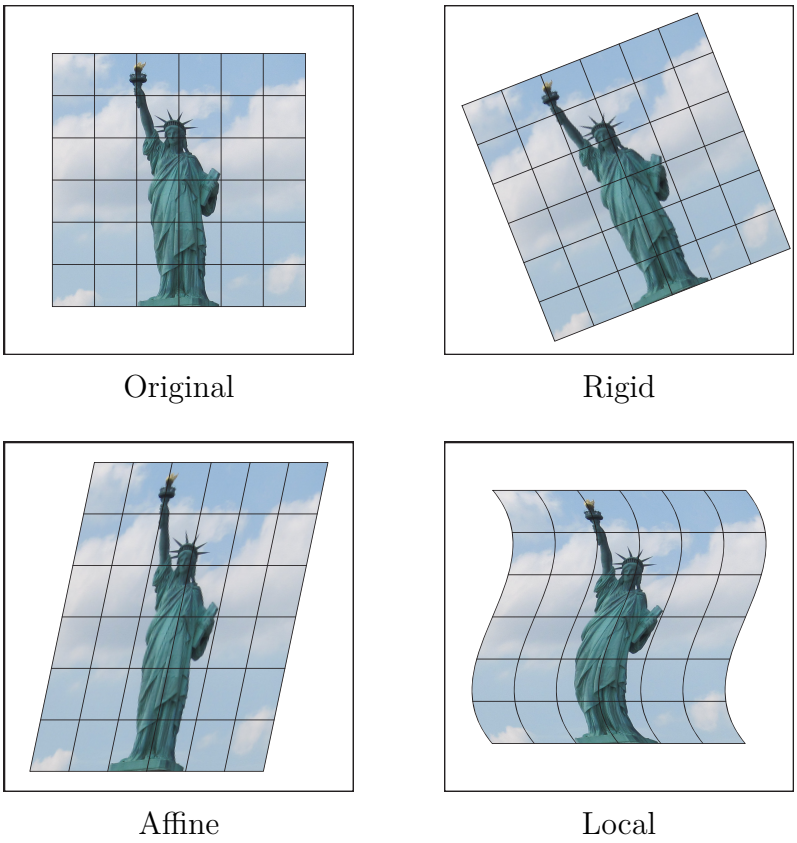


Figure 3.1: Different types of transformation illustrated on an image.

only allows for rotation and translation of the image, and so retains the shape of the image as well as the objects in it. It is mathematically defined as

$$\Psi(\mathbf{x}) = Q_\theta \mathbf{x} + \tau,$$

where $Q_\theta \in \mathbb{R}^{d\times d}$, such that $Q_\theta^\top Q_\theta = Q_\theta Q_\theta^\top = I_d$, is called the rotation matrix. Q_θ rotates the coordinates by angles $\theta \in [-\pi, \pi]^p$ around each axis ($p = 1$ for $d = 2$, and $p = 3$ for $d = 3$), and $\tau \in \mathbb{R}^d$ is the translation. For $d = 2$, the rotation matrix is given by the formula

$$Q_\theta = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}.$$

The affine transform is comprised of scaling and shear in addition to rotation and

translation. It is formally defined as

$$\Psi(\mathbf{x}) = C\mathbf{x} + \tau,$$

where $C \in \mathbb{R}^{d\times d}$ with $\det C > 0$.

Whereas the group transformations are global transforms that are unable to model local deformation, the deformable nonrigid transform is capable of it (see Figure 3.1). There are several deformable models such as wavelets [51], splines [24], and physically based models [9, 11], possessing varying degrees of freedom. In this thesis, physically based models are used and they will be discussed.

The general formulation for nonrigid registration can be written as:

$$\min_{\Psi}(\mathcal{D}[\Psi] + \beta\mathcal{E}[\Psi]), \tag{3.3}$$

where $\mathcal{E}[\Psi]$ is the regularization term, and $\beta \in \mathbb{R}^+$ is a user selected parameter. If we minimize the similarity measure alone by setting $\beta = 0$, then the minimization problem (3.3) is ill posed because there could be an infinite number of transforms that would minimize the objective function. Typically, the role of the regularizer is to restrict the transform to a diffeomorphism and to provide a unique solution. The nonrigid deformation is usually written in terms of a displacement field $\mathbf{r} : \Omega \to \mathbb{R}^d$ acting on the image, i.e., $\Psi(\mathbf{x}) = \mathbf{x} - \mathbf{r}$.

A common regularizer is the linear elastic potential energy [11, 31],

$$\mathcal{E}^{\text{elas}}[\mathbf{r}] \equiv \mathcal{P}[\mathbf{r}] = \int_\Omega \frac{\mu}{4} \sum_{j,k=1}^{d} \left(\partial_{x_j}\mathbf{r}_k + \partial_{x_k}\mathbf{r}_j\right)^2 + \frac{\lambda}{2}(\nabla.\mathbf{r})^2 d\mathbf{x}, \tag{3.4}$$

which models the deformation of an elastic body. The potential energy $\mathcal{P}[\mathbf{r}]$ is proportional to the deformation, so the objective function also tends to minimize deformation. The Lamé constants μ and λ are physical parameters, but for image registration they are usually set to some particular value. The minimizer of (3.3) with elastic regularizer (3.4) and SSD similarity (3.1), is computed by solving the Euler-Lagrange equation

$$\mu\triangle\mathbf{r} + (\lambda + \mu)\nabla(\nabla.\mathbf{r}) = \beta\left(A\left(\Psi(\mathbf{x})\right) - B(\mathbf{x})\right)\nabla A\left(\Psi(\mathbf{x})\right). \tag{3.5}$$

The deformation may also be modeled after a body which flows as a viscous fluid, [9]. Unlike the elastic model which is formulated as a minimization problem, the fluid

model is written as a system of two PDEs:

$$\begin{aligned} \mu \triangle \mathbf{v} + (\lambda + \mu) \nabla (\nabla . \mathbf{v}) &= \mathbf{b}, \\ \mathbf{v} &= \frac{\partial \mathbf{r}}{\partial t} + \nabla \mathbf{r}\ \mathbf{v}. \end{aligned} \tag{3.6}$$

The first equation is used to compute the velocity $\mathbf{v} : \Omega \rightarrow \mathbb{R}^d$ of the viscous fluid body under the applied body force $\mathbf{b} : \Omega \rightarrow \mathbb{R}^d$. The viscous fluid model is derived from the momentum equations of Navier Stokes. The force applied on the fluid is similar to the force applied on the elastic body (see [9]). Thus, in this case the body force is $\mathbf{b} = \beta (A(\Psi(\mathbf{x})) - B(\mathbf{x})) \nabla A(\Psi(\mathbf{x}))$, which is obtained from the variation of the similarity measure and present on the right hand side of (3.5). In other words, the elastic PDE in (3.5) is replaced with the fluid PDE in (3.6). The second equation gives the updated displacement of the body given the velocity. This model causes the body to flow like a viscous fluid with internal resistance to motion, but it does not restrict motion over a large number of time steps. In the case of registration, since the body force is nonlinear, the two equations are solved iteratively. When velocity is computed from the first equation, displacement is updated from the second, then the first equation is computed again with the updated force.

3.3 Segmentation

Segmentation is the task of finding regions in an image that belong to an object of interest. Segmentation methods are relevant for the purpose of object tracking, since they are used to detect objects in consecutive video frames. A vast number of methods have been developed for image segmentation which is not possible to discuss in short length. We list some important methods for segmentation which are classified broadly as

1. Clustering methods: These methods classify image pixels as belonging to clusters. Each cluster has shared features such as pixels of the same intensity or a particular range of intensity. K-means [28] or histogram [5] of the image are possible choices to compute the clusters. With K-means, pixels are assigned to K clusters so as to minimize the variance of the pixels from the cluster center. In histogram based methods, the peaks and valleys in the image histogram are detected, which correspond to clusters.

2. Region growing: In these methods, seeds, which are region labels, are spread over pixels in the image [1]. These seeded regions are then grown iteratively in its immediate neighborhood based on a growing criteria, until all pixels in the image have been assigned to images. The assignment criteria for instance could be whether the difference between the pixel intensity and the average region intensity lies within a specified threshold. Watershed transform [4] treats the image as a topographic surface. The altitude of the surface may be the grey level value or the magnitude of the image gradient. Regions are grown from the smallest value on the surface. Pixels with smaller values are assigned to existing regions before pixels with larger values. Pixels with a given value not assigned to any region are seeded as a new region. With this property, the watershed transform generates regions separated by lines along which the topographic values are high.

3. Active contour methods: The basic idea is to evolve a curve so that it moves to the boundary of the object. There are several models that fall into this category. One is the snakes model [21] which represents a curve as a parameterization over control points, such as spline. The curve is evolved until it lies over an edge in the image, where further evolution is stopped with the assumption that the object boundary has strong intensity gradients. Other methods involve evolving the curve until it segments regions in the image with some shared property, such as the average intensity of the region [13], texture [33], or a probabilistic measure on intensity [57]. Many active contour models [3, 13, 34, 33, 35, 55] represent the curve implicity as the zero level contour of level sets functions. Level set methods [42] are then used to evolve the curve by evolving the level set functions.

4. Graph partitioning: In these methods, for example one given in [43], the image pixels are modeled as nodes in an undirected graph. The weight of the edges may, for instance, specify the similarity between two nodes. The graph is then partitioned into clusters with each cluster representing a segmented region in the image.

Active contour methods are the most relevant to contour tracking of objects. In this section we discuss a particular contour segmentation model from Chan and Vese [13] and its formulation in a level set framework.

3.3.1 Chan-Vese Model

A widely used segmentation model is given by Chan and Vese [13]. Their model is used for segmenting uniform intensity regions in an image. Let a curve $\mathcal{C}$ be the boundary of an open bounded subset of the image domain such that it divides the image plane into two regions, i.e.

$$\begin{aligned} \omega \subsetneq \Omega, \qquad & \mathcal{C} = \partial\omega, \\ \text{inside}(\mathcal{C}) = \omega, \quad & \text{outside}(\mathcal{C}) = \Omega \setminus \bar{\omega}. \end{aligned}$$

The assumption is that the image $f : \Omega \to [0, 1]$ consists of two uniform intensity regions so that it may be described as

$$f(\mathbf{x}) = \begin{cases} c_1 & \mathbf{x} \in \omega \\ c_2 & \mathbf{x} \in \Omega \setminus \bar{\omega}. \end{cases}$$

Let the average intensity inside the two regions be denoted by

$$c_{\text{in}} = \frac{\int_\omega f(\mathbf{x}) d\mathbf{x}}{\int_\omega d\mathbf{x}} \qquad c_{\text{out}} = \frac{\int_{\Omega\setminus\bar{\omega}} f(\mathbf{x}) d\mathbf{x}}{\int_{\Omega\setminus\bar{\omega}} d\mathbf{x}} \quad .$$

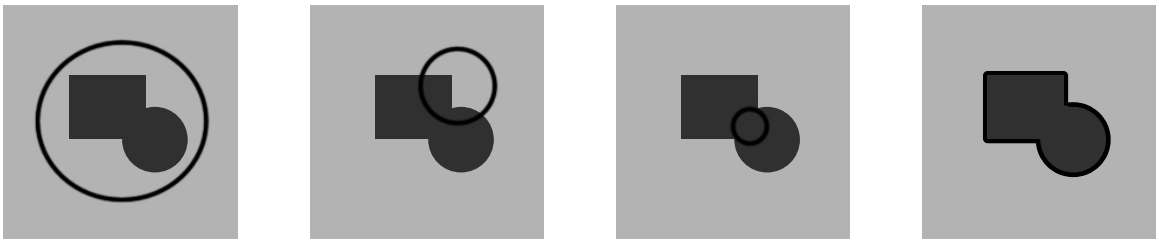
The deviation of segmented regions from a uniform intensity is measured by the terms

$$\begin{aligned} \mathcal{F}\left(\mathcal{C}, c_{\text{in}}, c_{\text{out}}\right) &= \mathcal{F}_1(\mathcal{C}, c_{\text{in}}) + \mathcal{F}_2(\mathcal{C}, c_{\text{out}}) \\ &= \int_\omega \left(f(\mathbf{x}) - c_{\text{in}}\right) d\mathbf{x} + \int_{\Omega\setminus\bar{\omega}} \left(f(\mathbf{x}) - c_{\text{out}}\right) d\mathbf{x}. \end{aligned} \tag{3.7}$$

The objective is to select a curve $\bar{\mathcal{C}}$ such that it minimizes the energy functional

$$\min_{\mathcal{C}} \left(\mathcal{F}\left(\mathcal{C}, c_{\text{in}}, c_{\text{out}}\right)\right) \approx 0 \approx \mathcal{F}_1(\bar{\mathcal{C}}) + \mathcal{F}_2(\bar{\mathcal{C}}).$$

In such a case the image will be partitioned into two regions, so that the average intensity of the two regions is equal to the intensity value defined over it, i.e., $c_{in} \approx c_1$ and $c_{out} \approx c_2$. The idea is illustrated in Figure 3.2 , where the two terms of the functional are evaluated on different curve configurations. The optimal curve is shown in Figure 3.2(d). Regularization terms may also be added to improve the well-posedness of the objective functional.



(a) (b) (c) (d)

Figure 3.2: Different possible curves are shown over an image, [13]. In (a) $\mathcal{F}_1 > 0, \mathcal{F}_2 \approx 0$, (b) $\mathcal{F}_1 > 0, \mathcal{F}_2 > 0$, (c) $\mathcal{F}_1 \approx 0, \mathcal{F}_2 > 0$, (d) $\mathcal{F}_1 \approx 0, \mathcal{F}_2 \approx 0$.

3.3.2 Level Set Representation

A closed curve $\mathcal{C}$ may be represented implicitly as the zero level curve Γ of a higher dimensional function $\Phi : \mathbb{R}^d \to \mathbb{R}$ i.e.,

$$\Gamma = \mathcal{C}, \qquad \Gamma = \{\mathbf{x} | \Phi(\mathbf{x}) = 0\},$$
$$\text{inside}(\mathcal{C}) = \{\mathbf{x} | \Phi(\mathbf{x}) < 0\}, \quad \text{outside}(\mathcal{C}) = \{\mathbf{x} | \Phi(\mathbf{x}) > 0\}.$$

$\Phi(\mathbf{x})$ is called the level set function, illustrated in Figure 3.3. Level set methods [42] are used to compute the optimal curve with implicit representation. With the level set method, the topology of the curve can split into different disconnected regions without any extra work. The contour and the regions are illustrated in Figure 3.4.

The curve $\mathcal{C}$ in (3.7) is represented by a level set function which leads to the functional

$$\min_{\Phi} \int_{\Omega} (f(\mathbf{x}) - c_{\text{in}})^2 (1 - H(\Phi(\mathbf{x}))) + (f(\mathbf{x}) - c_{\text{out}})^2 H(\Phi(\mathbf{x})) \, d\mathbf{x}, \tag{3.8}$$

where H is the Heaviside function

$$H(\Phi(\mathbf{x})) = \begin{cases} 0 & \Phi(\mathbf{x}) \leq 0, \\ 1 & \Phi(\mathbf{x}) > 0, \end{cases}$$

and c_{in} and c_{out} can be written in terms of $H(\Phi(\mathbf{x}))$ as

$$c_{\text{in}} = \frac{\int_{\Omega} f(\mathbf{x}) (1 - H(\Phi(\mathbf{x}))) \, d\mathbf{x}}{\int_{\Omega} (1 - H(\Phi(\mathbf{x}))) \, d\mathbf{x}}, \qquad c_{\text{out}} = \frac{\int_{\Omega} f(\mathbf{x}) H(\Phi(\mathbf{x})) \, d\mathbf{x}}{\int_{\Omega} H(\Phi(\mathbf{x})) \, d\mathbf{x}}.$$

Notice that, c_{in} is the average intensity of the region $\Phi(x) < 0$ while c_{out} is the average intensity of the region $\Phi(x) > 0$.

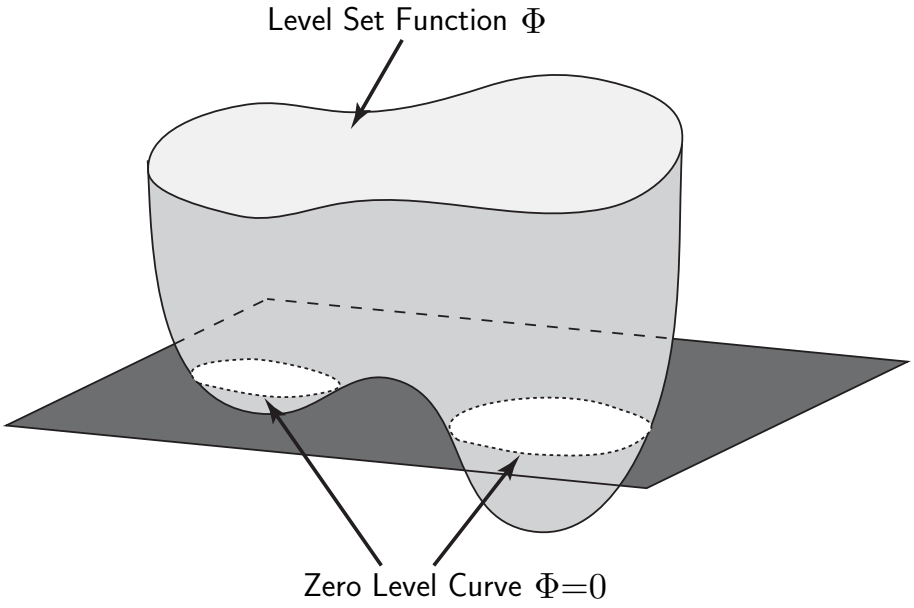


Figure 3.3: Level Set function with zero level contour.

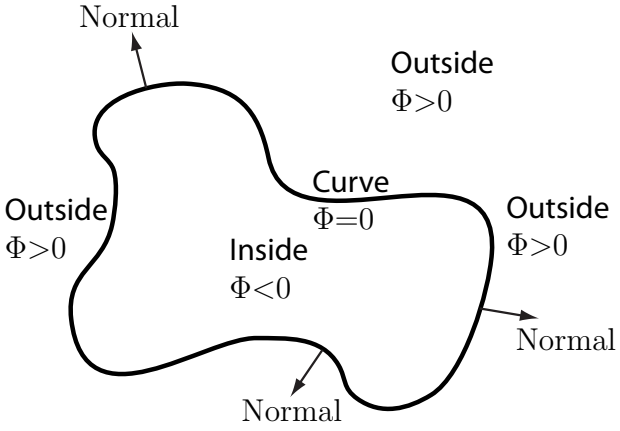


Figure 3.4: The contour plot of the level set function $\Phi(\mathbf{x})$ at the zero level set. The level set partitions the domain into inside and outside regions as indicated by the value of the function, [13].

Chapter 4

Tracking Model

In this chapter, the main idea for our tracking model is discussed. Motion based tracking of objects requires determining the motion that the object undergoes between a given video frame and the consecutive frame. Registration framework captures the "motion" between two given images. The framework can be modified so that it gives motion for individual objects between two given frames. Note that, this procedure is different from standard registration methods which deal with the entire images, whereas here it deals with objects in an image. The framework is capable of detecting occlusion and recovering the occluded region. We also discuss the idea of using segmentation measure to do tracking by embedding it in the tracking framework. This yields improved tracking results, since the segmentation measure is more robust to noise in the images and errors in tracking process.

4.1 Reconstruction Based Model

Given region, intensity, and motion information for all objects in the current frame, the next frame in a video can be reconstructed by applying the motion to the region and intensity templates of the object, and, combining all the transformed objects into a single image. The closer the reconstruction is to the actual frame, the more accurate will be the object motion that makes the reconstruction.

In case there are multiple objects in the video, the reconstruction process also needs to determine the depth ordering of the objects relative to each other. The

reason is that objects in real world scenario can get occluded in different ways. The objective is to capture the contour and obtain intensity template of the tracked object in both its occluded and unoccluded parts. If the occluded region is not tracked, tracking becomes more ill posed in later frames. For instance, when the occluded region becomes unoccluded, the tracking algorithm may incorrectly consider it to be a separate object or part of another object that lies in its vicinity. This affects the quality of reconstruction in later frames.

In general, tracking through occlusion is a highly ill-posed problem as the hidden part of the object can acquire any shape. Moreover, if the object is completely occluded, then theoretically it is not possible to determine the object's location and shape. Thus only partially occluded objects will be considered in this thesis, where information is available for the unoccluded part. Furthermore, it can be safely assumed for many cases, that, objects undergo a smooth deformation from frame to frame.

The reconstruction process takes into account the depth ordering, so that all regions in the reconstructed frame carry intensity information from the topmost lying object in that region. The occluded part of the object can be recovered by imposing additional conditions on the shape of the object. In the frame before which an object becomes occluded, the shape of the object is fully known and serves as its shape template. The motion of the object is constrained so that there are minimal changes in the shape of the object as compared to its shape template. This motion provides the shape and intensity information of the object in the occluded region.

We now develop representation for the video and objects in the scene. The frames in the video are denoted by $F(\mathbf{x}, t)$, where $\mathbf{x} \in [0,1]^d$, $d = 2$ is the spatial dimension of the image frame, $t = 1, \ldots, T$ denotes the current frame number, and $T \in \mathbb{N}$ is the total number of frames in the video. The current frame, called the initial frame, is denoted by $f(\mathbf{x}) \equiv F(\mathbf{x}, t)$ which gives the initial location of objects, and the next frame, called the target frame, as $g(\mathbf{x}) \equiv F(\mathbf{x}, t+1)$ which gives the final location of the objects. Suppose that there are $N \in \mathbb{N}$ objects to be tracked, which include both fixed and moving objects. Each object at current time t is represented by the following set of information:

$$\left.\begin{array}{ll} n & \text{object number} \\ \Phi^n(\mathbf{x}) & \text{level set function} \\ I^n(\mathbf{x}) & \text{intensity template} \end{array}\right\}, \quad \text{for } n = 1, 2, \ldots, N.$$

The level set function $\Phi^n(\mathbf{x})$ is used for representing the object domain Ω^n so that

$$\left.\begin{array}{ll} \Phi^n(\mathbf{x}) = 0, \text{ if } \mathbf{x} \in \partial\Omega^n & \text{object boundary} \\ \Phi^n(\mathbf{x}) < 0, \text{ if } \mathbf{x} \in \Omega^n & \text{object interior} \\ \Phi^n(\mathbf{x}) > 0, \text{ if } \mathbf{x} \notin \Omega^n & \text{object exterior} \end{array}\right\}, \text{ for } n = 1, 2, \ldots, N.$$

This is a convenient approach to keep track of the object region by evolving its level set from frame to frame. Also, when two objects overlap, the region of occlusion is simply given by the intersection of their respective level set functions in the object interior. It is also used for determining the region in the intensity template $I^n(\mathbf{x})$ which actually belongs to the object. The level set function thus constitutes an important part of the object model. The background is the region of the video frame that is outside of any objects in the scene. The background image of the scene is denoted by $B(\mathbf{x})$, $\mathbf{x} \notin \Omega^n$ for all $n = 1, 2, \ldots, N$. It is assumed to be available throughout the duration of tracking.

4.1.1 Rigid Tracking Model

First a simplified case is considered where the motion of the objects is assumed to be rigid in the video. Thus there is very little or no shape changes. In this case the objective for tracking simplifies to finding translation and rotation parameters. The approach is to reconstruct the target frame with information of objects from the intial frame. The reconstructed frame is then compared with the target frame in the video. The translation and rotation parameters are computed by minimizing the difference between the reconstructed frame and the target frame.

Consider a single object moved with translation parameters $\tau \in \mathbb{R}^d$ and rotated by $\theta \in [-\pi, \pi]$ around the center of mass. Let the rigid transform for the object be $\xi(\mathbf{x}) = Q_\theta \mathbf{x} + \tau$. The transformed object is then given by

$$\begin{cases} I_\xi(\mathbf{x}) \equiv I(\xi(\mathbf{x})) & \text{transformed intensity template,} \\ \Phi_\xi(\mathbf{x}) \equiv \Phi(\xi(\mathbf{x})) & \text{transformed level set.} \end{cases}$$

The reconstructed frame, $R(\mathbf{x}, \tau, \theta)$, is defined as

$$R(\mathbf{x}, \tau, \theta) = \begin{cases} B(\mathbf{x}) & \Phi_\xi(\mathbf{x}) > 0, \\ I_\xi(\mathbf{x}) & \Phi_\xi(\mathbf{x}) \leq 0. \end{cases} \tag{4.1}$$

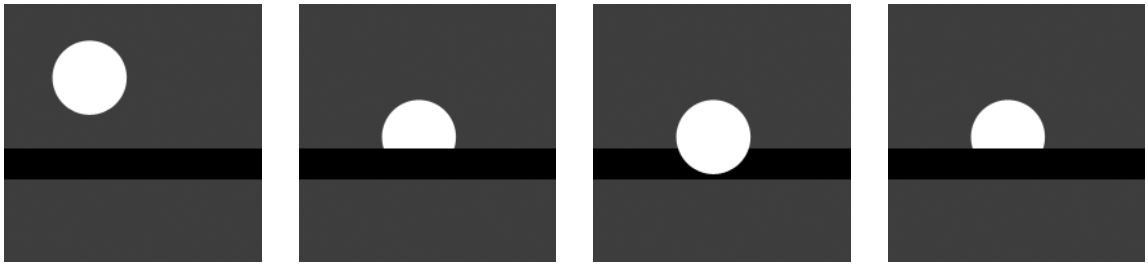
The reconstructed image has two regions - the unoccluded background, and, the transformed object. The tracking problem is then formulated as an unconstrained optimization problem:

$$\min_{\tau,\theta} \|R(\cdot,\tau,\theta) - g(\cdot)\|_2^2 . \tag{4.2}$$

The correct translation and rotation parameters will give a reconstruction similar to the target frame and will minimize the objective functional (4.2). The optimal parameters give an estimate of the motion of the object from the initial frame to the target frame.

4.1.1.1 Occlusion Tracking

In a case with a single tracked object, the object can become partially or completely hidden by a stationary object. Since the object is rigid, when the object is successfully located based on its unoccluded parts then the rest of the object's contour can also be determined. Whether the object is on top of or underneath another object also needs



(a) (b) (c) (d)

Figure 4.1: A moving circle becomes occluded by a fixed bar. Image (a) shows the initial frame, (b) the target frame, (c) the reconstruction with $z = (1, 2)$, and (d) the reconstruction with $z = (2, 1)$.

to be determined. Consider an example shown in Figure 4.1, which has a fixed object - a bar $I^1(\mathbf{x})$, along with the moving object - a circle $I^2(\mathbf{x})$. The two objects are each assigned a depth order by its position in a row vector z , where the position of the object's number in the vector determines its depth order. In general, the components of the vector, $z(i)$, $i = 1, \ldots, N$, represent the object number. By our convention, the object $I^{z(i)}(\mathbf{x})$ is occluded by object $I^{z(j)}(\mathbf{x})$ if $i < j$. The two possibilities for the example shown in Figure 4.1 are $z \in \{(1, 2), (2, 1)\}$. The reconstructed frame is then

defined as

$$R(\mathbf{x}, z, \tau, \theta) = \begin{cases} B(\mathbf{x}) & \Phi^1_{\xi^1}(\mathbf{x}) > 0 \text{ and } \Phi^2_{\xi^2}(\mathbf{x}) > 0, \\ I^{z(1)}_{\xi^{z(1)}}(\mathbf{x}) & \Phi^{z(1)}_{\xi^{z(1)}}(\mathbf{x}) \leq 0 \text{ and } \Phi^{z(2)}_{\xi^{z(2)}}(\mathbf{x}) > 0, \\ I^{z(2)}_{\xi^{z(2)}}(\mathbf{x}) & \Phi^{z(2)}_{\xi^{z(2)}}(\mathbf{x}) \leq 0. \end{cases}$$

The transform for the first object is the identity function $\xi^1(\mathbf{x}) = \mathbf{x}$ since the object is fixed, whereas for the second object it is $\xi^2(\mathbf{x}) = Q_\theta \mathbf{x} + \tau$. Now the reconstructed image has three regions - the unoccluded background, the unoccluded region of object $z(1)$, and the full object $z(2)$. Figure 4.1 shows the two possible reconstruction cases. Figure 4.1(c) corresponds to reconstruction with $z = (1, 2)$, and Figure 4.1(d) corresponds to reconstruction with $z = (2, 1)$. The latter reconstruction with $z = (2, 1)$ is the correct one. In presence of occlusion, depth ordering is necessary to give the correct reconstruction, and hence serves as an additional parameter. Thus, the minimization problem becomes:

$$\min_{z,\tau,\theta} \|R(\cdot, z, \tau, \theta) - g(\cdot)\|_2^2 . \tag{4.3}$$

4.1.1.2 Multiple Objects

The model in Section 4.1.1.1 can be extended to accommodate multiple moving objects. The idea is to simultaneously move all objects so that a correct reconstruction will be obtained similar to the target frame. At the same time, minimization over different depth orders produces a reconstruction that is closest to the target image.

For each object the translation parameter $\tau^n \in \mathbb{R}^d$, the rotation parameter $\theta^n \in [-\pi, \pi]$, and the depth order $z \in P(1, 2, \ldots, n)$, where $P(1, 2, \ldots, n)$ represents the set of permutations, are the reconstruction parameters. For N objects there will be a total of $2N$ translation parameters, N rotation parameters, and a set of $N!$ different permuted depth orderings. The unconstrained minimization problem is

$$\min_{z,\boldsymbol{\tau},\boldsymbol{\theta}} \|R(\cdot, z, \boldsymbol{\tau}, \boldsymbol{\theta}) - g(\cdot)\|_2^2 , \tag{4.4}$$

where $\boldsymbol{\tau} \equiv \left\{\tau^1, \tau^2 \ldots, \tau^N\right\}$, $\boldsymbol{\theta} \equiv \left\{\theta^1, \theta^1, \ldots, \theta^N\right\}$, the piecewise function

$$R\left(\mathbf{x}, z, \boldsymbol{\tau}, \boldsymbol{\theta}\right) = \begin{cases} B(\mathbf{x}) & \Phi^n_{\xi^n}(\mathbf{x}) > 0 \; (n = 1, 2, \ldots, N), \\ I^{z(1)}_{\xi^{z(1)}}(\mathbf{x}) & \Phi^{z(1)}_{\xi^{z(1)}}(\mathbf{x}) \leq 0 \text{ and } \Phi^{z(n)}_{\xi^{z(n)}}(\mathbf{x}) > 0 \; (n = 2, \ldots, N), \\ \vdots & \vdots \\ I^{z(m)}_{\xi^{z(m)}}(\mathbf{x}) & \Phi^{z(m)}_{\xi^{z(m)}}(\mathbf{x}) \leq 0 \text{ and } \Phi^{z(n)}_{\xi^{z(n)}}(\mathbf{x}) > 0 \; (n = m+1, \ldots, N), \\ \vdots & \vdots \\ I^{z(N)}_{\xi^{z(N)}}(\mathbf{x}) & \Phi^{z(N)}_{\xi^{z(N)}}(\mathbf{x}) \leq 0, \end{cases} \tag{4.5}$$

$\Phi^n_{\xi^n}(\mathbf{x}) \equiv \Phi^n\left(\xi^n(\mathbf{x})\right)$, $I^n_{\xi^n}(\mathbf{x}) \equiv I^n\left(\xi^n(\mathbf{x})\right)$, and the rigid transform $\xi^n(\mathbf{x}) = Q_{\theta^n}\mathbf{x} + \tau^n$. The objective functional is first minimized for object motion parameters for all given depth orders and then the minimum is taken over different depth orders.

4.1.2 Nonrigid Tracking Model

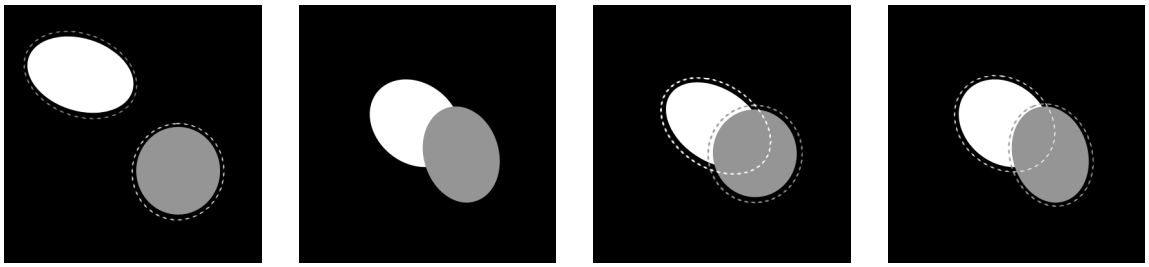
Uptil now only objects that appear rigid in the video have been considered. In practice, however, there would be deformations in the object. To make the tracking more general, the objects are allowed to deform as well. The rigid model for single object tracking is modified so that a displacement $\mathbf{r} : \Omega \to \mathbb{R}^d$ is defined for every point of the object domain. The problem can then be solved similar to (4.2), by solving for the displacement field in the minimization. The tracking problem is formulated as the following minimization problem:

$$\min_{\mathbf{r}} \left(\alpha \left\|R(\cdot, \mathbf{r}) - g(\cdot)\right\|_2^2 + \mathcal{E}[\mathbf{r}]\right), \tag{4.6}$$

where $\mathcal{E}[\mathbf{r}]$ is the regularization term for the reconstruction, and $R\left(\mathbf{x}, \mathbf{r}\right)$ is as defined in (4.1) but replacing translation τ and rotation θ with displacement $\mathbf{r}(\mathbf{x})$, $\Phi_\xi(\mathbf{x})$ with $\Phi_\Upsilon(\mathbf{x}) \equiv \Phi\left(\Upsilon(\mathbf{x})\right)$ and $I_\xi(\mathbf{x})$ with $I_\Upsilon(\mathbf{x}) \equiv I\left(\Upsilon(\mathbf{x})\right)$, and, the nonrigid transform $\Upsilon(\mathbf{x}) = \mathbf{x} - \mathbf{r}(\mathbf{x})$. There are several choices for the regularizer. Two possible candidates are $\mathcal{E}[\mathbf{r}] = \mathcal{E}^{elas}[\mathbf{r}]$ (3.4) and $\mathcal{E}[\mathbf{r}] = \mathcal{E}^{vf}[\mathbf{r}]$ (3.6).

Dealing with occlusion is similar to the technique presented in Section 4.1.1.1. Multiple objects tracking is also similar to Section 4.1.1.2. In this case, we compute the sum of the regularization terms for all objects

$$\min_{z, \vec{\mathbf{r}}} \left(\alpha \left\|R\left(\cdot, z, \vec{\mathbf{r}}\right) - g(\cdot)\right\|_2^2 + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n]\right), \tag{4.7}$$



(a) (b) (c) (d)

Figure 4.2: Nonrigid tracking of two objects is shown in two consecutive frames. The dotted lines indicate the boundary contour for the object. The images (a) and (b) show the initial frame and the target frame, respectively. Image (c) is the reconstruction with rigid model, and (d) the reconstruction with the nonrigid model.

where $\vec{\mathbf{r}} \equiv \left\{\mathbf{r}^1, \mathbf{r}^2, \ldots, \mathbf{r}^N\right\}$, $\mathcal{E}[\mathbf{r}^n]$ is the regularization term for object n, $R\left(\cdot, z, \vec{\mathbf{r}}\right)$ is as defined in (4.5) but replacing translation τ^n and rotation θ^n with displacement $\mathbf{r}^n \equiv \mathbf{r}^n(\mathbf{x})$, $\Phi^n_{\xi^n}(\mathbf{x})$ with $\Phi^n_{\Upsilon^n}(\mathbf{x}) \equiv \Phi^n\left(\Upsilon^n(\mathbf{x})\right)$ and $I^n_{\xi^n}(\mathbf{x})$ with $I^n_{\Upsilon^n}(\mathbf{x}) \equiv I^n\left(\Upsilon^n(\mathbf{x})\right)$, the nonrigid transform $\Upsilon^n(\mathbf{x}) = \mathbf{x} - \mathbf{r}^n(\mathbf{x})$, z is the depth order, and $\alpha \in \mathbb{R}^+$ is the weight of the similarity measure. The nonrigid model is now able to capture the deformable object contours. An example is shown in Figure 4.2. The rigid reconstructed image in Figure 4.2(c) does not minimize the functional in (4.7), since it leaves a residual in the first term of (4.7). The deformation of the object can be captured by applying the nonrigid model. The reconstruction as shown in Figure 4.2(d) does not leave any residual in the similarity measure since the first term of (4.7) is approximately zero. This reconstruction is the optimal solution and it matches the expected results.

The reconstructed image does not change if the object shape is changed in the occluded region. This is apparent from the definition of the reconstruction (4.5) where the occluded parts of the object are not present in the reconstucted image. Hence, the similarity measure also does not change with object shape change in the occluded region. The value of the regularization term, however, increases proportional to the deformation of the object. Minimizing the functional hence prevents large shape changes. The shape of the object in the occluded region is thus retained, while still allowing shape changes in the unoccluded region.

We note that level set segmentation model (3.8) for tracking requires computing the level set function over the entire image. If the image is large then this computation

becomes quite expensive. In the nonrigid model (4.7), the displacement field needs to be computed over the objects only. Thus, the computation time depends only on the size and resolution of the object in the image. For the rigid model, there are only two unknowns for each object. Our model, therefore, has fewer unknowns and computationally less expensive.

4.1.3 Combined Rigid and Nonrigid Model

The nonrigid model by itself, however, may not capture occlusion. Consider the example in Figure 4.3. The circle gets occluded by the bar in the target frame. If the nonrigid model is applied, then the obtained contour, as shown in Figure 4.3(c), does not include the circle's occluded part. This is because the similarity measure in (4.7) remains unchanged if the object shape is changed inside the region occluded by the bar. The reconstructed image remains the same regardless of object shape in the occluded part. On the other hand, the value of the regularization term generally increases with greater deformation in the object shape. The expected solution does not coincide with the minimum of (4.7), as it yields an overall higher value of the functional as opposed to the one shown in Figure 4.3(c). The minimization process will thus deform the object to match the unoccluded part only.

To address this issue, the object's motion is decomposed into rigid and nonrigid components. If we apply the rigid model in (4.4), and perform an intermediate rigid reconstruction, as in Figure 4.3(d), the global rigid motion of the object can be obtained. Even though at this stage no nonrigid motion is accounted for, it gives an initial estimate for the location of the object in the target frame. More importantly, this step gives an estimate of the occluded part of object as well as the depth ordering. The depth order and rigid transform parameters $(z, \boldsymbol{\tau}, \boldsymbol{\theta})$ are fixed and then the nonrigid model (4.7) is applied to obtain the final reconstruction, as shown in Figure 4.3(e). The objects deform until their unoccluded regions match with those in the target frame. The regularizer produces a smooth and continuous displacement field over the object. As a result, the occluded part of the object is also slightly deformed.

Let us denote the optimal rigid transform parameters for each object, obtained from solving (4.4), are $(\hat{\tau}^n, \hat{\theta}^n)$, and the optimal depth order is $\hat{z}$. The optimal transform is then $\hat{\mathbf{x}} = \hat{\xi}^n(\mathbf{x}) = Q_{\hat{\theta}^n}\mathbf{x} + \hat{\tau}^n$. These paramters are then fixed in the nonrigid step and the displacement field is computed. The transform Υ^n in (4.4) is replaced with

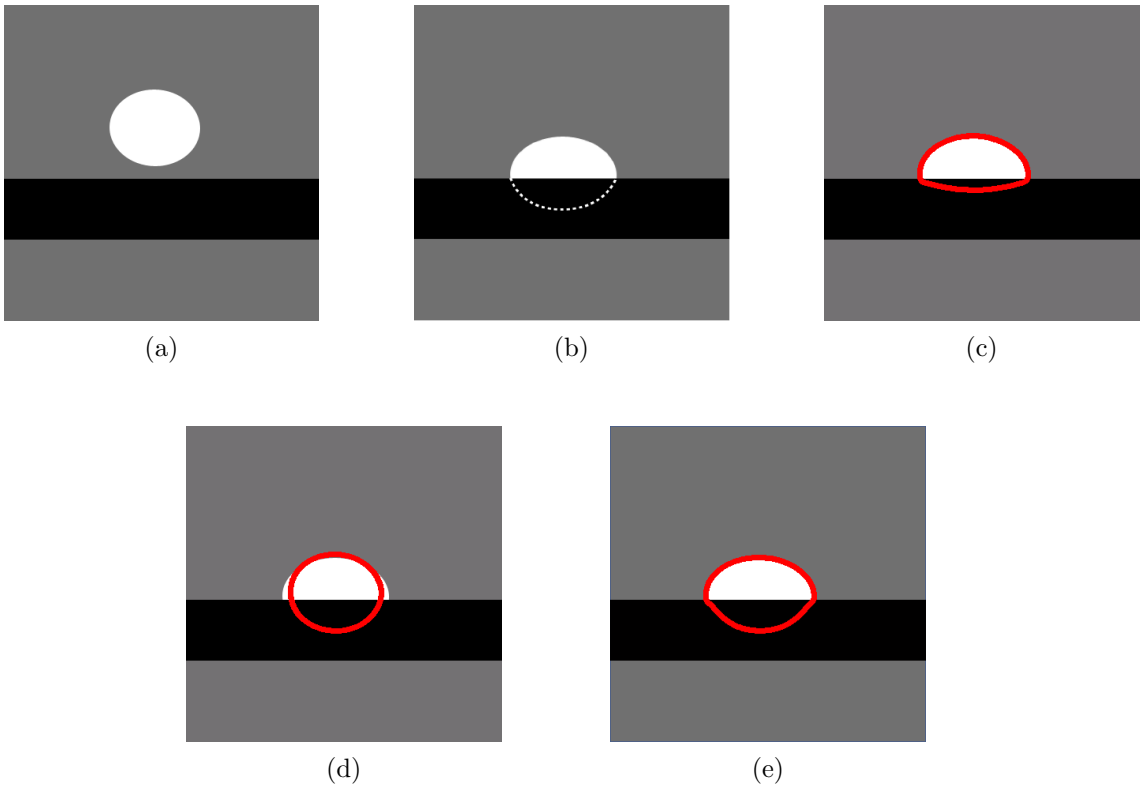


Figure 4.3: The images (a) and (b) show the initial frame and the target frame, respectively. Image (c) shows the reconstruction using the nonrigid model only. Images (d) and (e) show the intermediate reconstruction with the rigid model and the combined rigid and nonrigid model, respectively.

the combined transform

$$\begin{aligned} \Psi^n(\mathbf{x}) &= \Upsilon^n(\hat{\xi}^n(\mathbf{x})) \\ &= \hat{\mathbf{x}} - \mathbf{r}^n(\hat{\mathbf{x}}). \end{aligned} \tag{4.8}$$

This transform is composed over the optimal transform from the rigid step, so the remaining unknown $\mathbf{r}^n(\hat{\mathbf{x}})$ is computed from the nonrigid model, which is now stated as

$$\min_{\vec{\mathbf{r}}} \ \alpha \mathcal{D}^{rec}[\vec{\mathbf{r}}] + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n]. \tag{4.9}$$

The measure of quality of reconstruction $\mathcal{D}^{rec}[\vec{\mathbf{r}}]$ is denoted in short by $\mathcal{D}^{rec}$ and defined

as

$$\mathcal{D}^{rec} \equiv \left\| R\left(\cdot, \hat{z}, \hat{\boldsymbol{\tau}}\hat{\boldsymbol{\theta}}, \vec{\mathbf{r}}\right) - g(\cdot) \right\|_2^2. \tag{4.10}$$

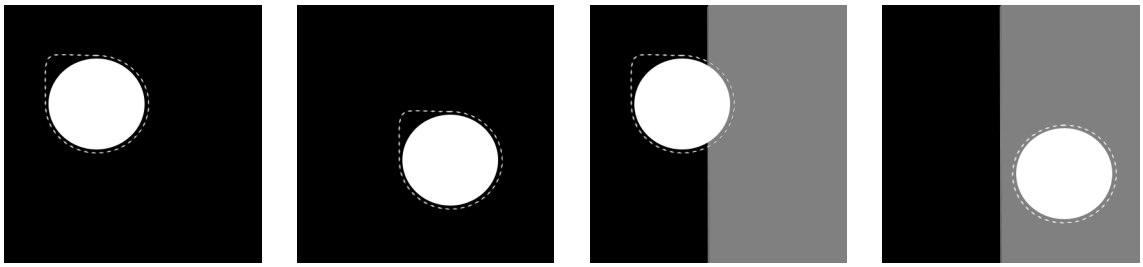
Note that the tracking model given in (4.9) is computed via a piecewise composition of the reconstructed image, where each piece corresponds to different objects or the background. The reconstructed image (4.5) can be expressed in terms of the object intensity images $I^n_{\Psi^n}(\mathbf{x})$ and the level set functions with $H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right)$. Then the reconstruction measure (4.10) is written equivalently as:

$$\begin{aligned}\mathcal{D}^{rec} = &\sum_{n=1}^{N} \int_{\Omega} \left(I^n_{\Psi^n}(\mathbf{x}) - g(\mathbf{x})\right)^2 \left(1 - H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right)\right) \prod_{k \epsilon K} H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right) d\mathbf{x} \\ &+ \int_{\Omega} \left(B(\mathbf{x}) - g(\mathbf{x})\right)^2 \prod_{n=1}^{N} H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right) d\mathbf{x},\end{aligned} \tag{4.11}$$

where $K = \{z(m+1), \ldots, z(N)\}$, s.t. $z(m) = n$. The benefit of using this formulation as opposed to the one in (4.10) is that, in the former, the domain of integration is the whole domain of the image, whereas, in the latter, the domains are the boundaries of the unoccluded parts of the objects. Differentiating through the integral of the measure, in the former case, requires computing the derivatives of the boundary of the integral explicitly as the integral domain is variable with respect to object motion. In the latter case, derivatives of the integral boundary are zero since it is fixed with respect to object domain. Thus it is easier to compute the derivatives of the integrals.

4.2 Segmentation Measure

While the L^2 norm on the difference image (4.10) works well in many cases, it may not give the right object boundary when the background has a uniform intensity. Consider Figure 4.4(a) and Figure 4.4(b). They show two consecutive frames where a white object moves from left to right on a black background. Suppose there is a small error in tracking the object boundary in the first frame as shown in Figure 4.4(a). This error will persist in the following frames. The reason is that the similarity measure (4.2) in the incorrectly captured region is zero because the intensity of the reconstructed image (4.1) is the same as the background intensity in the target frame. The minimization process would not deform the object in this region as the similarity



(a) (b) (c) (d)

Figure 4.4: The images (a) and (b) show consecutive frames with a uniform background. The error is retained in the tracking in this case. The other two images (c) and (d) are consecutive frames with a varying background. The error in the tracked contour is corrected.

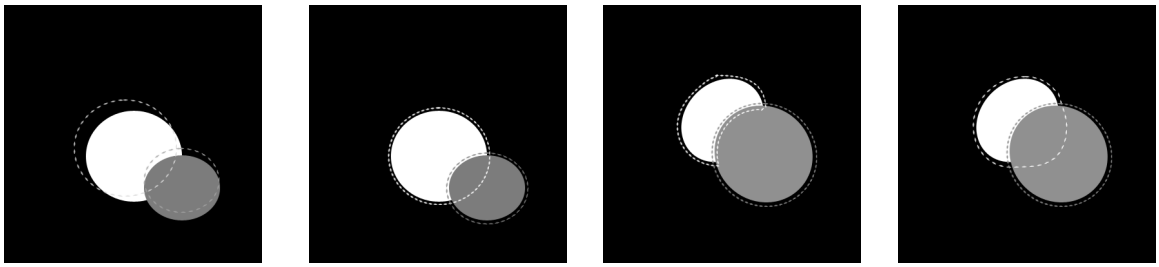
has already reached the minimum possible value. The nonrigid model will not deform the incorrect object contour to the correct one, as seen in Figure 4.4(b). This type of error can progressively accumulate over several frames as the object changes shape, and become difficult to remove. The error will only be self corrected if the background intensity in the vicinity of the object changes in a future frame as seen in Figure 4.4(c) and Figure 4.4(d). In this case, the intensity of the incorrect region is black which does not match the gray background of the target. Thus the contour will shrink to minimize the similarity measure against the target frame.

In order to address this issue, we propose the use of a segmentation energy measure (3.8), in place of the usual L^2 measure. In this approach the benfits of registration and segmentation are combined together. Only the object displacement $\mathbf{r}(\mathbf{x})$ needs to be computed, rather than the entire level set function $\Phi(\mathbf{x})$ as is the case in segmentation model. At the same time we are able to use the more robust measure provided by the segmentation model.

First, consider the case of a single rigid object. In our new registration based model, the objective functional given in (3.8) is changed to

$$\begin{aligned}\min_{\tau,\theta} \int_{\Omega} \Big(&(g(\mathbf{x}) - c_{\text{in}})^2 \left(1 - H\left(\Phi_{\xi}(\mathbf{x})\right)\right) \\ &+ (g(\mathbf{x}) - c_{\text{out}})^2 H\left(\Phi_{\xi}(\mathbf{x})\right)\Big)\, d\mathbf{x},\end{aligned} \tag{4.12}$$

where $\xi(\mathbf{x}) = Q_\theta \mathbf{x} + \tau$. The main difference between (3.8) and (4.12) is that in the



(a) (b) (c) (d)

Figure 4.5: Images (a) and (b) demonstrate the case of rigid object motion in the target frame. Images (c) and (d) are target frames for nonrigid object motion. The images (a) and (c) show the contours with non optimal solution, whereas (b) and (d) show optimal solutions.

former is to find the level set function that segments the object, and in the latter, the objective is to find the rigid transform of the object from the initial frame to the target frame. The contour $\Phi_\xi(\mathbf{x})$ will minimize the functional when the displacement applied to it provides an accurate fit to the boundary of the object. If the error as shown in Figure 4.4(a) was present in the past frame, then in the future frames, the error will be corrected because the similarity measure will not be minimized until the contour $\Phi_\xi(\mathbf{x})$ separates the object and the background into regions of uniform intensity. Compared to (4.2), this model is more robust to tracking errors in past frames.

We now consider the two object case. The original Chan-Vese model [13] assumes the boundaries of objects are disjoint, and none of the boundaries are occluded. Here, the energy functional is generalized so that it will capture occlusion. Consider the nonrigid case of two moving objects; see Figure 4.5. For the depth order $z = (1, 2)$, we define the new energy functional as follows:

$$\begin{aligned}\min_{z,\boldsymbol{\tau},\boldsymbol{\theta}} \int_\Omega \Big(& \Big(g(\mathbf{x}) - c_{\text{in}}^1\Big)^2 \Big(1 - H\left(\Phi_{\xi^1}^1(\mathbf{x})\right)\Big) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right) \\ & + \Big(g(\mathbf{x}) - c_{\text{in}}^2\Big)^2 \Big(1 - H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\Big) \\ & + \left(g(\mathbf{x}) - c_{\text{out}}\right)^2 H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\Big) d\mathbf{x},\end{aligned} \tag{4.13a}$$

where

$$c_{\text{in}}^1 = \frac{\int_\Omega g(\mathbf{x}) \left(1 - H\left(\Phi_{\xi^1}^1(\mathbf{x})\right)\right) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right) d\mathbf{x}}{\int_\Omega \left(1 - H\left(\Phi_{\xi^1}^1(\mathbf{x})\right)\right) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right) d\mathbf{x}},$$
$$c_{\text{in}}^2 = \frac{\int_\Omega g(\mathbf{x}) \left(1 - H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\right) d\mathbf{x}}{\int_\Omega \left(1 - H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\right) d\mathbf{x}}.$$

c_{in}^1 is the average of the intensity for the unoccluded region of object 1, and c_{in}^2 is the average of the intensity of object 2 which is unoccluded in this case. The first term in (4.13a) computes the sum of the difference between the average intenisty of object 1 and the target frame over unoccluded regions of object 1. The second term is the sum of the difference between the average intensity of object 2 and the target frame. The last term is the sum over the difference outside all object regions; i.e. the background. If the depth order is $z = (2, 1)$, then the energy functional becomes:

$$\begin{aligned} \min_{z,\boldsymbol{\tau},\boldsymbol{\theta}} \int_\Omega \Big(& \left(g(\mathbf{x}) - c_{\text{in}}^1\right)^2 \left(1 - H\left(\Phi_{\xi^1}^1(\mathbf{x})\right)\right) \\ & + \left(g(\mathbf{x}) - c_{\text{in}}^2\right)^2 H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) \left(1 - H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\right) \\ & + \left(g(\mathbf{x}) - c_{\text{out}}\right)^2 H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right) \Big) d\mathbf{x}, \end{aligned} \tag{4.13b}$$

where

$$c_{\text{in}}^1 = \frac{\int_\Omega g(\mathbf{x}) \left(1 - H\left(\Phi_{\xi^1}^1(\mathbf{x})\right)\right) d\mathbf{x}}{\int_\Omega \left(1 - H\left(\Phi_{\xi^1}^1(\mathbf{x})\right)\right) d\mathbf{x}},$$
$$c_{\text{in}}^2 = \frac{\int_\Omega g(\mathbf{x}) H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) \left(1 - H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\right) d\mathbf{x}}{\int_\Omega H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) \left(1 - H\left(\Phi_{\xi^2}^2(\mathbf{x})\right)\right) d\mathbf{x}}.$$

Here the average intensities are defined similar to the previous case, but with object 1 considered to be unoccluded and object 2 considered occluded. For both cases,

$$c_{\text{out}} = \frac{\int_\Omega g(\mathbf{x}) H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right) d\mathbf{x}}{\int_\Omega H\left(\Phi_{\xi^1}^1(\mathbf{x})\right) H\left(\Phi_{\xi^2}^2(\mathbf{x})\right) d\mathbf{x}}.$$

The new energy functional considers occlusion in contrast to the Chan-Vese model, which does not. The functional computes the average intensity only on the unoccluded parts of the objects. Moreover, the difference between average intensity and image

frame is taken over the unoccluded regions of the objects. If the average intensity were to include both of the occluded and unoccluded regions, then the average intensity would be different from the object intensity; see Figure 4.5(a). The difference between the average intensity and the intensity of region inside the contour, for both the objects, is nonzero at all points inside the contour. The difference between the average intensity of the region outside the contours and the background intensity is also nonzero at all points in the outside region. Hence, the contours in Figure 4.5(a) do not minimize the model.

The functionals for the different depth orders generally give different similarity measure values for the same translation and rotation, when the objects overlap. For instance in Figure 4.5(b), the first term in the functional (4.13b) is nonzero because it includes the occluded part. The average intensity of object 1, which is a shade of grey instead of white, when subtracted from the target frame leaves a residual. The functional (4.13a) does not include the occluded part of the object in the first term, so it has all the terms zero. The contours with $z = \{1, 2\}$ are thus the optimal solution.

The energy functional (4.13a) can be generalized to N objects. Let m be the position of the object n in the depth order vector i.e. $z(m) = n$, and $K = \{z(m+1), \ldots, z(N)\}$ be the set of all the object numbers that lie on top of object n. Then the tracking model is given by

$$\begin{aligned}\min_{z,\boldsymbol{\tau},\boldsymbol{\theta}} \sum_{n=1}^{N} \int_{\Omega} (g(\mathbf{x}) - c_{\text{in}}^{n})^2 \left(1 - H\left(\Phi_{\xi^n}^{n}(\mathbf{x})\right)\right) \prod_{k \epsilon K} H\left(\Phi_{\xi^k}^{k}(\mathbf{x})\right) d\mathbf{x} \\ + \int_{\Omega} (g(\mathbf{x}) - c_{\text{out}})^2 \prod_{n=1}^{N} H\left(\Phi_{\xi^n}^{n}(\mathbf{x})\right) d\mathbf{x},\end{aligned} \tag{4.14}$$

where

$$\begin{aligned} c_{\text{in}}^{n} &= \frac{\int_{\Omega} g(\mathbf{x}) \left(1 - H\left(\Phi_{\xi^n}^{n}(\mathbf{x})\right)\right) \prod_{k \epsilon K} H\left(\Phi_{\xi^k}^{k}(\mathbf{x})\right) d\mathbf{x}}{\int_{\Omega} \left(1 - H\left(\Phi_{\xi^n}^{n}(\mathbf{x})\right)\right) \prod_{k \epsilon K} H\left(\Phi_{\xi^k}^{k}(\mathbf{x})\right) d\mathbf{x}}, \\ c_{\text{out}} &= \frac{\int_{\Omega} g(\mathbf{x}) \prod_{n=1}^{N} H\left(\Phi_{\xi^n}^{n}(\mathbf{x})\right) d\mathbf{x}}{\int_{\Omega} \prod_{n=1}^{N} H\left(\Phi_{\xi^n}^{n}(\mathbf{x})\right) d\mathbf{x}}. \end{aligned} \tag{4.15}$$

The extension of (4.14) to combine rigid and nonrigid case is straightforward. The displacement field is added as a parameter in addition to translation and rotation, so

the model becomes

$$\min_{\vec{\mathbf{r}}} \ \alpha \mathcal{D}^{seg}[\vec{\mathbf{r}}] + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n]. \tag{4.16}$$

The segmentation measure $\mathcal{D}^{seg}[\vec{\mathbf{r}}]$ is denoted in short by $\mathcal{D}^{seg}$ and defined as

$$\begin{aligned} \mathcal{D}^{seg} \equiv & \sum_{n=1}^{N} \int_{\Omega} \left(g(\mathbf{x}) - c_{\text{in}}^n\right)^2 \left(1 - H\left(\Phi_{\Psi^n}^n(\mathbf{x})\right)\right) \prod_{k \epsilon K} H\left(\Phi_{\Psi^k}^k(\mathbf{x})\right) d\mathbf{x} \\ & + \int_{\Omega} \left(g(\mathbf{x}) - c_{\text{out}}\right)^2 \prod_{n=1}^{N} H\left(\Phi_{\Psi^n}^n(\mathbf{x})\right) d\mathbf{x}. \end{aligned} \tag{4.17}$$

c_{in}^n and c_{out} are similar to the definition in the rigid case, but replacing $\Phi_{\xi^n}^n(\mathbf{x})$ with $\Phi_{\Psi^n}^n(\mathbf{x}) \equiv \Phi^n\left(\Psi^n(\mathbf{x})\right)$. The transform $\Psi^n(\mathbf{x})$ is as defined in (4.8) using the optimal parameters obtained from the rigid step.

We note that in segmentation based trackers, occlusion is detected and handled separately from the segmentation. For instance, the tracking method given by Yilmaz and Shah [55] first segments objects with a segmentation functional, and then separately detects and recovers occlusion. Our model uses the segmentation functional as a similarity measure. The minimization process handles the occlusion and a separate procedure is not required.

For the rigid case, once the unoccluded part of the object is correctly tracked, then the estimate for the contour in the occluded part is obtained as well. This may not be the case for nonrigid tracking. Consider Figure 4.5(c) and Figure 4.5(d). The combined rigid and nonrigid model (Section 4.1.3) is applied to track the white and gray objects. Suppose the rigid tracking step has been performed. Thus part of the contour for the white object has already moved to the occluded region. Now the nonrigid model is applied using the new energy functional (4.16). The contours in both Figure 4.5(c) and Figure 4.5(d) would give rise to the same minimum similarity measure. The reason is that the average intensity is computed over the unoccluded region, so it will not be changed by a change in object shape in the occluded region. Moreover, the difference between the average intensity and the target frame is computed over the unoccluded region only, so the difference measure will also not change. In other words, the energy functional would not differentiate the two cases. In fact, one can select any contour for the white object in the occluded region and get the same value for the segmentation energy. The minimization process, however, will lead to the results in Figure 4.5(d). Also, the regularization term allows for only small deformations. Thus, after the rigid

tracking step, the contour in the occluded region will remain more or less unchanged, while the nonrigid model captures the contour of the occluded part.

The segmentation measure devised here is based on the assumption that the images in the video are composed of uniform intensity regions. This assumption is similar to the one in Chan-Vese model, where the image is composed of two distinct uniform intensities. However, more sophisticated measures may easily be devised from other level set segmentation methods, such as Yilmaz et al. [55], which can detect regions based on appearance characteristics.

4.3 Color Images

Color images can be represented in different color spaces. Red-Blue-Green (RGB) is the most common color representation on digital images. Alternative representations include Hue-Saturation-Lightness (HSL), Hue-Saturation-Value (HSV), and Luma-Chrominane (Y'UV) among others. Tracking in color image sequences can be done using any of the color spaces. We use RGB format for our model with the color channels $q \in \mathcal{Q} = \{\text{red, green, blue}\}$. Each video frame is represented by the three intensity images $\langle F^q(\mathbf{x}, t)\rangle$, target frame by $\langle g^q(\mathbf{x})\rangle$, object intensities by $\langle I^{q,n}(\mathbf{x})\rangle$, and the reconstructed image denoted by $\langle R^q(\mathbf{x}, z, \boldsymbol{\tau}, \boldsymbol{\theta}, \vec{\mathbf{r}})\rangle$. The objects can be tracked independently in the three component images. This results in a set of three functionals - one for each color channel, which are as following:

$$\begin{aligned}
&\min_{\vec{\mathbf{r}}} \left(\alpha \left\| R^{\text{red}}(\cdot, z, \boldsymbol{\tau}, \boldsymbol{\theta}, \vec{\mathbf{r}}) - g^{\text{red}}(\cdot) \right\|_2^2 + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n] \right), \\
&\min_{\vec{\mathbf{r}}} \left(\alpha \left\| R^{\text{green}}(\cdot, z, \boldsymbol{\tau}, \boldsymbol{\theta}, \vec{\mathbf{r}}) - g^{\text{green}}(\cdot) \right\|_2^2 + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n] \right), \\
&\min_{\vec{\mathbf{r}}} \left(\alpha \left\| R^{\text{blue}}(\cdot, z, \boldsymbol{\tau}, \boldsymbol{\theta}, \vec{\mathbf{r}}) - g^{\text{blue}}(\cdot) \right\|_2^2 + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n] \right).
\end{aligned} \tag{4.18}$$

The three objective functionals are solved simultaneously. Since the system is overdetermined, least squares method is used to yield a single displacement field for each object. The approach is to minimize over the sum of the norm of the difference image for the three color channels, which can be written as:

$$\min_{\vec{\mathbf{r}}} \left(\alpha \sum_{q \in \mathcal{Q}} \left\| R^q\left(\cdot, z, \boldsymbol{\tau}, \boldsymbol{\theta}, \vec{\mathbf{r}}\right) - g^q(\cdot) \right\|_2^2 + \sum_{n=1}^{N} \mathcal{E}[\mathbf{r}^n] \right) \tag{4.19}$$

The minimum of the summed similarity measure may not coincide with the minimum of similarity in individual color channels. The approach works because it gives the best estimate of the actual displacement in the sense of least squares fitting. The segmentation based measure can be extended in a similar manner by summing the measure in the RGB color channels. Note that we have left out describing the rigid tracking step but the objective functionals for it are similar to those discussed in 4.1.1.2. Also, the reconstruction measures in (4.19), are restated in the form of (4.26) when it is actually implemented.

In the alternate color image representation, the HSV color space can be used to gain control over certain aspects of tracking. For instance, if the object changes intensity from frame to frame because of shadows, this change is reflected mostly in the Value component of the HSV space. In order to make tracking less sensitive to shadows and illumination, the weight of the Value component may be set to zero or a small number, and tracking can be done with the Hue-Saturation components alone.

4.4 Minimizers for Tracking Model

In this section, we will compute the equations for minimizing the tracking models (4.9), (4.16), and (4.19). In this first part, the gradient of the models are derived with respect to the rigid transform parameters. In the second part, the Euler-Lagrange equations are obtained for solving the nonrigid tracking step.

For the sake of brevity, we will first define some terms that will be used in the rest of this thesis, before deriving the minimizers of the tracking model. Let

$$\begin{aligned}
K_1 &= \{1, \ldots, N\} \setminus \{n\}, \\
K_2 &= \{z(1), \ldots, z(m-1)\}, \text{ s.t. } z(m) = n, \\
K_3 &= \{z(m+1), \ldots, z(N)\}, \text{ s.t. } z(m) = n, \\
K_4 &= \{z(m+1), \ldots, z(N)\}, \text{ s.t. } z(m) \in K_2, \\
K_5 &= K_4 \setminus \{n\},
\end{aligned} \tag{4.20}$$

where K_1 is the set of object indices except for the object n, K_2 is the set of object indices occluded by object n, K_3 is the set of object indices occluding object n, K_4 and K_5 is the set of object indices that occlude an object that is itself occluded by object n, with K_4 including object n and K_5 excluding it. The following are also defined:

$$\begin{aligned}
\mathfrak{T}_1 &= \delta\left(\Phi^n_{\chi^n}(\mathbf{x})\right)\nabla\Phi^n_{\chi^n}(\mathbf{x})\frac{\partial\chi^n}{\partial s}, & \mathfrak{T}_2 &= \prod_{k\epsilon K_3} H\left(\Phi^k_{\chi^k}(\mathbf{x})\right), \\
\mathfrak{T}_3 &= \left(1 - H\left(\Phi^k_{\chi^k}(\mathbf{x})\right)\right)\prod_{\bar{k}\epsilon K_5} H\left(\Phi^{\bar{k}}_{\chi^{\bar{k}}}(\mathbf{x})\right), & \mathfrak{T}_4 &= \prod_{k\in K_1} H\left(\Phi^k_{\chi^k}(\mathbf{x})\right), \\
\mathfrak{T}_5 &= \left(1 - H\left(\Phi^n_{\chi^n}(\mathbf{x})\right)\right)\mathfrak{T}_2, & \mathfrak{T}_6 &= \prod_{n=1}^{N} H\left(\Phi^n_{\chi^n}(\mathbf{x})\right), \\
\mathfrak{T}_7 &= \left(1 - H\left(\Phi^k_{\chi^k}(\mathbf{x})\right)\right)\prod_{\bar{k}\epsilon K_4} H\left(\Phi^{\bar{k}}_{\chi^{\bar{k}}}(\mathbf{x})\right).
\end{aligned} \tag{4.21}$$

Here $\delta(x)$ is the Dirac delta function, $\nabla I^n_{\chi^n}(\mathbf{x})$ and $\nabla\Phi^n_{\chi^n}(\mathbf{x})$ are the gradients of transformed intensity image and level set function. In the rigid tracking step, we denote the tracking parameter and transform to be $s^n \in \{\tau^n, \theta^n\}$ and $\chi^n(\mathbf{x}) = \xi^n(\mathbf{x})$ respectively, and, in the nonrigid tracking step, to be $s^n = \mathbf{r}^n(\hat{\mathbf{x}})$ and $\chi^n(\mathbf{x}) = \Psi^n(\mathbf{x})$ respectively .

4.4.1 Rigid Step - Gradient

For the rigid tracking step, the reconstruction measure is similar to (4.11), but replacing $\Psi^n(\mathbf{x})$ with $\xi^n(\mathbf{x})$. Levenberg-Marquardt algorithm is used to solve the rigid model which requires the similarity measure be expressed as a sum of the squares terms. So, the terms associated with the level set functions also squared. This does not change the minimum of the model and allows use of nonlinear least squares optimization. It is then written as:

$$\mathcal{D}^{rec} = \sum_{n=1}^{N}\int_\Omega \left(\mathfrak{D}^n_1\right)^2 d\mathbf{x} + \int_\Omega \left(\mathfrak{D}_2\right)^2 d\mathbf{x}, \tag{4.22}$$

where

$$\mathfrak{D}^n_1 = \left(I^n_{\xi^n}(\mathbf{x}) - g(\mathbf{x})\right)\mathfrak{T}_5, \quad \mathfrak{D}_2 = \left(B(\mathbf{x}) - g(\mathbf{x})\right)\mathfrak{T}_6. \tag{4.23}$$

Let $s^n \in \{\tau^n, \theta^n\}$ belong to the set of rigid transform parameters for object n. The gradient of the terms (4.23) with respect to the model parameters are

$$\begin{aligned}\frac{\partial \mathfrak{D}_1^n}{\partial s^n} &= \mathfrak{T}_1 \left(I_{\xi^n}^n(\mathbf{x}) - g(\mathbf{x})\right) \mathfrak{T}_2 + \nabla I_{\chi^n}^n(\mathbf{x}) \frac{\partial \xi^n}{\partial s^n} \mathfrak{T}_5, \\ \frac{\partial \mathfrak{D}_2}{\partial s^n} &= \mathfrak{T}_1 \left(B(\mathbf{x}) - g(\mathbf{x})\right) \mathfrak{T}_4.\end{aligned} \tag{4.24}$$

The derivative of the spatial transform is

$$\frac{\partial \xi^n}{\partial s^n} = \begin{cases} 1 & s^n = \tau^n, \\ \dfrac{\partial Q_{\theta^n}}{\partial \theta^n}\mathbf{x} & s^n = \theta^n, \end{cases} \tag{4.25}$$

where the derivative of the rotation matrix for two dimensional case, i.e. $d = 2$, is

$$\frac{\partial Q_{\theta^n}}{\partial \theta^n} = \begin{bmatrix} -\sin\theta & \cos\theta \\ -\cos\theta & -\sin\theta \end{bmatrix}.$$

The segmentation measure (4.17) is similarly written as

$$\mathcal{D}^{seg} = \sum_{n=1}^{N} \int_\Omega \left(\mathfrak{D}_2^n\right)^2 d\mathbf{x} + \int_\Omega \left(\mathfrak{D}_4\right)^2 d\mathbf{x}, \tag{4.26}$$

where

$$\mathfrak{D}_3^n = \left(g(\mathbf{x}) - c_{\text{in}}^n\right) \mathfrak{T}_5, \quad \mathfrak{D}_4 = \left(g(\mathbf{x}) - c_{\text{in}}\right) \mathfrak{T}_6. \tag{4.27}$$

The gradient of the terms (4.27) are

$$\begin{aligned}\frac{\partial \mathfrak{D}_3^n}{\partial s^n} &= -\mathfrak{T}_1 \left(g(\mathbf{x}) - c_{\text{in}}^n\right) \mathfrak{T}_2 - \frac{\partial c_{\text{in}}^n}{\partial s^n} \mathfrak{T}_5, \\ \frac{\partial \mathfrak{D}_4}{\partial s^n} &= \mathfrak{T}_1 \left(g(\mathbf{x}) - c_{\text{out}}\right) \mathfrak{T}_4 - \frac{\partial c_{\text{out}}}{\partial s^n} \mathfrak{T}_6,\end{aligned} \tag{4.28}$$

where for $k \in K_2$

$$\frac{\partial c_{\text{in}}^n}{\partial s^n} = -\frac{\int_\Omega g(\mathbf{x}) \mathfrak{T}_1 \mathfrak{T}_2 d\mathbf{x}}{\int_\Omega \mathfrak{T}_5 d\mathbf{x}} + \frac{\int_\Omega g(\mathbf{x}) \mathfrak{T}_5 d\mathbf{x} \int_\Omega \mathfrak{T}_1 \mathfrak{T}_2 d\mathbf{x}}{\left(\int_\Omega \mathfrak{T}_5 d\mathbf{x}\right)^2}, \tag{4.29a}$$

$$\frac{\partial c_{\text{in}}^k}{\partial s^n} = \frac{\int_\Omega g(\mathbf{x}) \mathfrak{T}_1 \mathfrak{T}_3 d\mathbf{x}}{\int_\Omega \mathfrak{T}_7 d\mathbf{x}} - \frac{\int_\Omega g(\mathbf{x}) \mathfrak{T}_7 d\mathbf{x} \int_\Omega \mathfrak{T}_1 \mathfrak{T}_3 d\mathbf{x}}{\left(\int_\Omega \mathfrak{T}_7 d\mathbf{x}\right)^2}, \tag{4.29b}$$

$$\frac{\partial c_{\text{out}}}{\partial s^n} = \frac{\int_\Omega g(\mathbf{x})\mathfrak{T}_1\mathfrak{T}_4 d\mathbf{x}}{\int_\Omega \mathfrak{T}_6 d\mathbf{x}} - \frac{\int_\Omega g(\mathbf{x})\mathfrak{T}_6 d\mathbf{x} \int_\Omega \mathfrak{T}_1\mathfrak{T}_4 d\mathbf{x}}{\left(\int_\Omega \mathfrak{T}_6 d\mathbf{x}\right)^2}. \tag{4.29c}$$

The gradient of the rigid tracking model gives the direction for steepest ascent and descent with respect to the rigid tracking parameters. It is required for computing the minimizers when using least squares based optimization algorithms, such as Gauss-Newton and Levenberg-Marquardt method. In the next chapter, the gradient will be used for solving the rigid tracking step; Section 5.3.

4.4.2 Nonrigid Step - Euler-Lagrange Equations

For the regularization term in the tracking model we can either use the linear elastic model (3.4) or substitute the regularizer with the viscous fluid model (3.6). Although it is possible to use either of these options, we choose to use the viscous fluid regularizer for reasons which we discuss here in short. The linear elastic model allows only small linear deformations in the body. The elastic regularization term used in the tracking model is, therefore, restrictive in the shape change of the tracked object. This can cause the tracked contour to not match the actual contour of the object. The fluid model, on the other hand, describes motion of viscous fluids and so allows nonlinear deformation in the object.

For the nonrigid step, we derive the Euler-Lagrange equations (refer to [50] for details) of (4.9) with respect to displacement $\mathbf{r}^n \equiv \mathbf{r}^n(\hat{\mathbf{x}})$ and keeping the rigid parameters fixed as obtained from the rigid step. The elastic regularizer is used in the derivation of Euler-Lagrange equations. Similar to the fluid registration model (see 3.2.4), we replace the elastic PDE with the viscous fluid PDE, while keeping the body force the same. For the sake of brevity, we skip listing the elastic equations and directly provide the fluid equations. Objects are then tracked by solving the following

equations:

$$\begin{aligned}
\mu\triangle\mathbf{v}^1+(\lambda+\mu)\,\nabla\left(\nabla.\mathbf{v}^1\right)&=\alpha\frac{\partial\mathcal{D}^{rec}}{\partial\mathbf{r}^1},\quad \mathbf{v}^1=\frac{\partial\mathbf{r}^1}{\partial\bar{t}}+\nabla\mathbf{r}^1\;\mathbf{v}^1,\\
\mu\triangle\mathbf{v}^2+(\lambda+\mu)\,\nabla\left(\nabla.\mathbf{v}^2\right)&=\alpha\frac{\partial\mathcal{D}^{rec}}{\partial\mathbf{r}^2},\quad \mathbf{v}^2=\frac{\partial\mathbf{r}^2}{\partial\bar{t}}+\nabla\mathbf{r}^2\;\mathbf{v}^2,\\
&\vdots\qquad\qquad\qquad\vdots\\
\mu\triangle\mathbf{v}^N+(\lambda+\mu)\,\nabla\left(\nabla.\mathbf{v}^N\right)&=\alpha\frac{\partial\mathcal{D}^{rec}}{\partial\mathbf{r}^N},\quad \mathbf{v}^N=\frac{\partial\mathbf{r}^N}{\partial\bar{t}}+\nabla\mathbf{r}^N\;\mathbf{v}^N,
\end{aligned}\tag{4.30}$$

where $\bar{t}$ is the time for computing the velocity from the displacement. This time $\bar{t}$ is different from the video time t which is used to denote the frame number. Each object has associated with it two equations, for a total of $2N$ equations for N objects. The first one is for computing the velocity of the object and the second one is for computing the updated displacement from the velocity. The body force $\mathbf{b}(\mathbf{x})$ is given by the variation of the similarity measure $\mathcal{D}^{rec}$ (4.26):

$$\begin{aligned}
\frac{\partial\mathcal{D}^{rec}}{\partial\mathbf{r}^n}=&\mathfrak{T}_1\Bigg(-\left(I^n_{\Psi^n}(\mathbf{x})-g(\mathbf{x})\right)^2\mathfrak{T}_2+\sum_{k\epsilon K_2}\left(I^k_{\Psi^k}(\mathbf{x})-g(\mathbf{x})\right)^2\mathfrak{T}_3\\
&+\left(B(\mathbf{x})-g(\mathbf{x})\right)^2\mathfrak{T}_4\Bigg)+2\left(I^n_{\Psi^n}(\mathbf{x})-g(\mathbf{x})\right)\nabla I^n_{\Psi^n}(\mathbf{x})\frac{\partial\Psi^n}{\partial\mathbf{r}^n}\mathfrak{T}_5
\end{aligned}\tag{4.31}$$

where the terms are as defined previously in (4.20) and (4.36). The derivative of the spatial transform used in the definitions is

$$\frac{\partial\Psi^n}{\partial\mathbf{r}^n}=-1,$$

as it follows from straightforward differentiation. The solution of (4.30) is not exactly the solution of (4.9). However, by using the fluid model, large nonlinear deformation in the object shape can be captured as opposed to when using the elastic model and tracking results will be better.

There are two components of the force in (4.31). One is on the boundaries of the objects arising from the first three terms, and the other is inside the unoccluded region of the objects coming from the last term. Force is nonzero where there is a difference in intensity between the target frame and the reconstruction, and, zero in regions where there is no difference. The second component of force is scaled by the gradient of the intensity image of the object so that stronger gradients will give rise to

higher magnitude forces. The edges corresponding to the gradients inside the object, implicitly act as features to be registered. A sharper edge means more information is available in its region with which to drive the registration. Note that the force is zero inside the occluded region since all the terms in (4.31) are computed on the unoccluded part of the object. This keeps the shape of the occluded regions intact while still deforming the unoccluded regions. The fluid PDE (3.6) produces a smooth deformation. The occluded parts of the object will thus deform sufficiently with the rest of the object so as to result in a smooth object boundary.

Similarly, the Euler-Lagrange equations for the segmentation measure model (4.16) are:

$$\begin{aligned}
\mu\triangle\mathbf{v}^1+(\lambda+\mu)\,\nabla\left(\nabla.\mathbf{v}^1\right)&=\alpha\frac{\partial\mathcal{D}^{seg}}{\partial\mathbf{r}^1},\quad \mathbf{v}^1=\frac{\partial\mathbf{r}^1}{\partial\bar{t}}+\nabla\mathbf{r}^1\;\mathbf{v}^1,\\
\mu\triangle\mathbf{v}^2+(\lambda+\mu)\,\nabla\left(\nabla.\mathbf{v}^2\right)&=\alpha\frac{\partial\mathcal{D}^{seg}}{\partial\mathbf{r}^2},\quad \mathbf{v}^2=\frac{\partial\mathbf{r}^2}{\partial\bar{t}}+\nabla\mathbf{r}^2\;\mathbf{v}^2,\\
&\vdots\qquad\qquad\vdots\\
\mu\triangle\mathbf{v}^N+(\lambda+\mu)\,\nabla\left(\nabla.\mathbf{v}^N\right)&=\alpha\frac{\partial\mathcal{D}^{seg}}{\partial\mathbf{r}^N},\quad \mathbf{v}^N=\frac{\partial\mathbf{r}^N}{\partial\bar{t}}+\nabla\mathbf{r}^N\;\mathbf{v}^N.
\end{aligned}\tag{4.32}$$

where $\mathcal{D}^{seg}$ is the segmentation measure given in (4.17). From calculus of variations, we obtain

$$\begin{aligned}
\frac{\partial\mathcal{D}^{seg}}{\partial\mathbf{r}^n}=&\mathfrak{T}_1\Bigg(-\left(g(\mathbf{x})-c_{\text{in}}^n\right)^2\mathfrak{T}_2+\sum_{k\epsilon K_2}\left(g(\mathbf{x})-c_{\text{in}}^k\right)^2\mathfrak{T}_3\\
&+\left(g(\mathbf{x})-c_{\text{out}}\right)^2\mathfrak{T}_4\Bigg)-2\left(g(\mathbf{x})-c_{\text{in}}^n\right)\frac{\partial c_{\text{in}}^n}{\partial s}\mathfrak{T}_5\\
&-\sum_{k\epsilon K_2}2\left(g(\mathbf{x})-c_{\text{in}}^k\right)\frac{\partial c_{\text{in}}^k}{\partial s}\mathfrak{T}_7-2\left(g(\mathbf{x})-c_{\text{out}}\right)\frac{\partial c_{\text{out}}}{\partial\mathbf{r}^n}\mathfrak{T}_6,
\end{aligned}\tag{4.33}$$

where the terms are as defined previously in (4.20), (4.36), (4.29a), (4.29b) and (4.29c), but with $s^n=\mathbf{r}^n$ and the variation of the spatial transform

$$\frac{\partial\Psi^n}{\partial\mathbf{r}^n}=-1.$$

For color images, using the model in (4.19) and the definition of the reconstruction measure in (4.35), the corresponding Euler-Lagrange equations are similar to (4.32) with

$$\frac{\partial \mathcal{D}^{rec}}{\partial \mathbf{r}^n} = \sum_{q \in \mathcal{Q}} \Bigg(\mathfrak{T}_1 \Bigg(- \left(I^{q,n}_{\Psi^n}(\mathbf{x}) - g^q(\mathbf{x}) \right)^2 \mathfrak{T}_2 + \sum_{k \epsilon K_2} \left(I^{q,k}_{\Psi^k}(\mathbf{x}) - g^q(\mathbf{x}) \right)^2 \mathfrak{T}_3 + \left(B^q(\mathbf{x}) - g^q(\mathbf{x}) \right)^2 \mathfrak{T}_4 \Bigg) + 2 \left(I^{q,n}_{\Psi^n}(\mathbf{x}) - g^q(\mathbf{x}) \right) \nabla I^{q,n}_{\Psi^n}(\mathbf{x}) \frac{\partial \Psi^n}{\partial \mathbf{r}^n} \mathfrak{T}_5 \Bigg),$$

where $\mathcal{Q} = \{\text{red, green, blue}\}$. Note that the forces produced by the three color channels are simply added to give a net force at a given point. The color channel in which the difference between the reconstruction and the target image is higher will influence the net force more. The Euler-Lagrange equations thus tend to reduce the measure more in that channel as opposed to the others. The extension of the segmentation model to color images is similar to the extension in reconstruction model. The forces are computed in each channel by (4.33) and summed to give a net force. The net force is then the body force used in the equations (4.32).

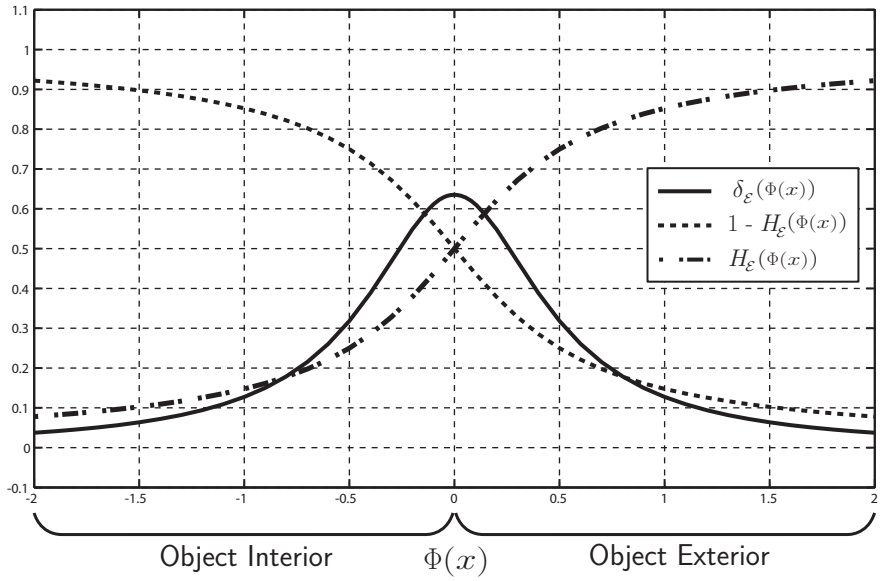
4.5 Force on Object Boundary

The Dirac delta function $\delta(x)$ present in the variation of the reconstruction and segmentation measures is not possible to compute numerically as it is defined to be infinity at a single point $x = 0$. Instead a regularized version $\delta_\varepsilon(x)$ is used as an approximation, such that $\lim_{\varepsilon \to 0} \delta_\varepsilon(x) = \delta(x)$. The Heaviside step function defined as $H(x) = \int_{-\infty}^{\infty} \delta(x)\, dx$, is also regularized to correspond with the use of regularized delta function. One choice for the functions are a $C^\infty(\Omega)$ regularization given in [13]

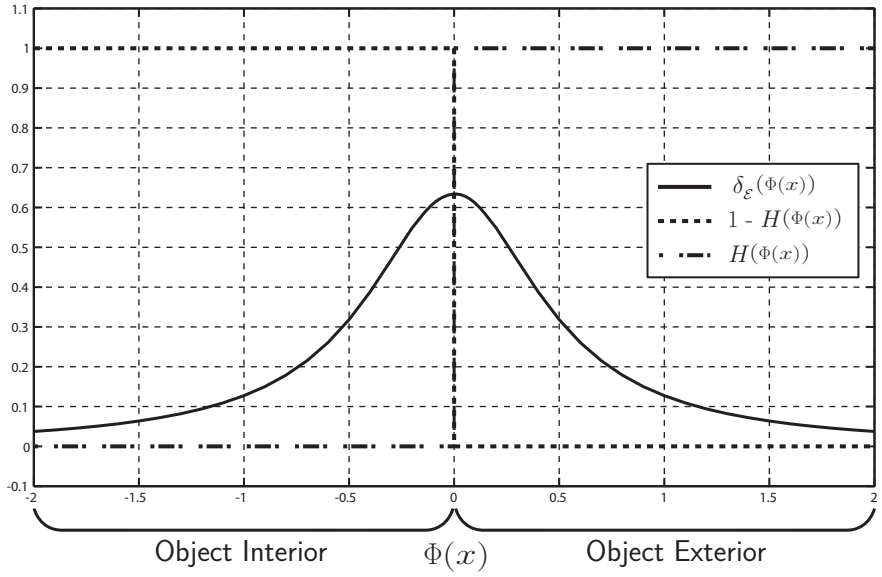
$$\begin{aligned} H_\varepsilon(x) &= \frac{1}{2}\left(1 + \frac{2}{\pi}\tan^{-1}\left(\frac{x}{\varepsilon}\right)\right), \\ \delta_\varepsilon(x) &= \frac{\varepsilon/\pi}{x^2 + \varepsilon^2}. \end{aligned} \tag{4.34}$$

Notice that the function $H_\varepsilon(\Phi(x)) > 0,$ for $\Phi(x) < 0$ (Figure 4.6(a)) as opposed to $H(\Phi(x)) = 0,$ for $\Phi(x) < 0$ (Figure 4.6(b)), and $\delta_\varepsilon(\Phi(x)) \not> 0,$ for $\Phi(x) \neq 0$ (Figure 4.6(a)), compared with $\delta(\Phi(x)) = 0,$ for $\Phi(x) \neq 0$.

The approximate step and delta functions are used in the terms of (4.21). Thus, the object intensity images present in all the terms of (4.31), are evaluated inside



(a)



(b)

Figure 4.6: (a) The plot of the regularized step function and delta function. (b) The step function with regularized delta function. The object inside and outside region is defined by the level set function.

the object boundary as well as outside. However, object intensity is only defined inside the object region, with its outside undefined. This may give incorrect tracking results. Consider an object being tracked in Figure 4.7, with the initial frame in (4.7a) and target frame in (4.7b). The object is shown overlayed on a pixel grid, with the boundary lying between two pixels. Though not shown in the figure, the inside region of the object intensity template is white and outside is black, while the background intensity is black. Two boundary forces act on the object according to (4.31). The term $\mathrm{T1} = \left(I^n_{\Psi^n}(\mathbf{x}) - g(\mathbf{x})\right)^2 \mathfrak{T}_2$ is the magnitude of the inward pulling force, and, $\mathrm{T2} = \left(B(\mathbf{x}) - g(\mathbf{x})\right)^2 \mathfrak{T}_4$ is the magnitude of the outward pulling force, and the net force magnitude is $-\mathrm{T1} + \mathrm{T2}$. These terms are evaluated at pixels on both sides of the object boundary since the regularized step and delta functions are used. The figures show only the force acting on one pixel at a time, but it is the same for all pixels lying around the top boundary. The net force on the object boundary in Figure 4.7(a), evolves the object outwards to result in the object in Figure 4.7(b). It should stop evolving at this stage as it has deformed into the object in the target frame. This, however, is not the case, since the net force on the object boundary in Figure 4.7(b) is nonzero due to $\mathrm{T2} > 0$. The boundary force evolves the object further outwards to result in the object in Figure 4.7(c), which has pixels captured from the background. In Figure 4.7(c), the net force is still nonzero because $\mathrm{T1} > 0$, but this time it pulls the object inwards resulting in the same configuration as Figure 4.7(b). The contour thus keeps oscillating between these positions. Eventually, when the solution to the system in (4.30) converges, some pixels from the background are often incorrectly classified as part of the object. Such an issue does not arise in segmentation based measures since segmentation models do not use object intensity templates. Instead they work by taking into account explicitly whether a pixel in a target frame belongs to a particular object or the background. Thus for the segmentation models the regularized step and delta functions are used.

The boundary force on the object depends on how the object intensity templates are defined. But since it is not apparent what the object intensity should be in regions outside its boundary, it may not make sense to evaluate the image difference terms of (4.31) outside the defined object region. The model is modified so that the terms do not get evaluated in regions of undefined intensity. The reconstruction measure in the

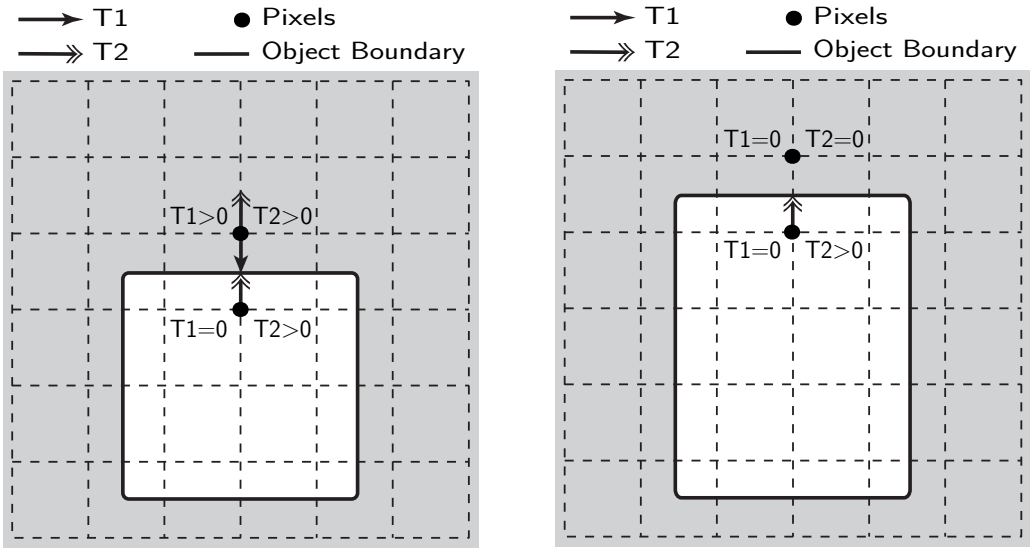


Figure 4.7: Evolution of the object contour in a scene with a single object using model terms (4.21) with regularized step and delta functions. (a) The initial frame, (b) the target frame, and (c) the incorrect object contour, show different stages in the evolution of object shape.

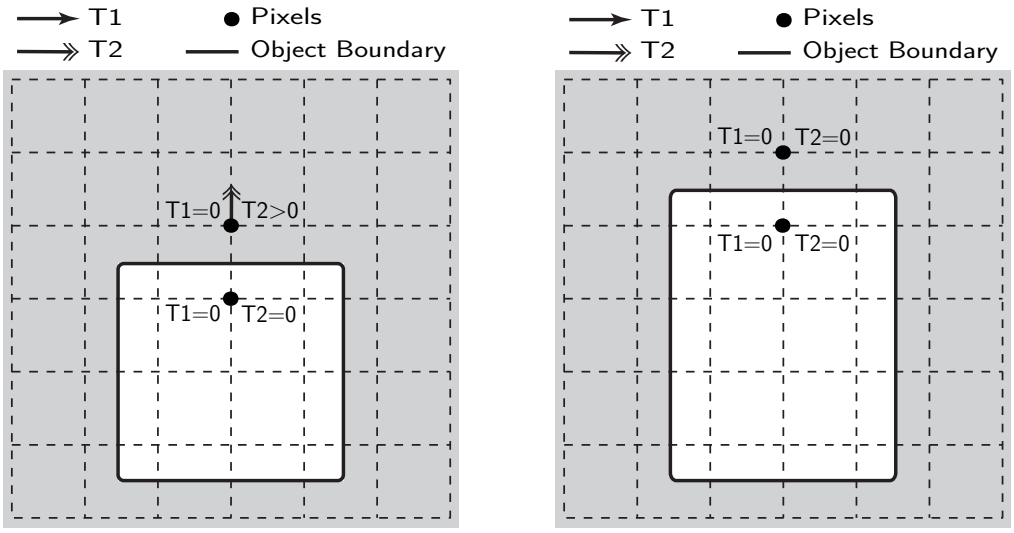
model is then modified to

$$\begin{aligned}\mathcal{D}^{rec} = &\sum_{n=1}^{N} \int_{\Omega} \left(I^n_{\Psi^n}(\mathbf{x}) - g(\mathbf{x})\right)^2 \left(1 - H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right)\right)^2 \prod_{k \epsilon K} H\left(\Phi^k_{\Psi^k}(\mathbf{x})\right)^2 d\mathbf{x} \\ &+ \int_{\Omega} \left(B(\mathbf{x}) - g(\mathbf{x})\right)^2 \prod_{n=1}^{N} H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right)^2 d\mathbf{x},\end{aligned} \tag{4.35}$$

where the inside and outside region terms have been squared compared to (4.31). Its gradient with respect to rigid parameters and variation with respect to displacement is given by (4.31) respectively with the terms in it redefined as

$$\begin{aligned}\mathfrak{T}_1 &= 2\delta_{\varepsilon}\left(\Phi^n_{\Psi^n}(\mathbf{x})\right) \nabla \Phi^n_{\Psi^n}(\mathbf{x}) \frac{\partial \Psi^n}{\partial s}, \\ \mathfrak{T}_2 &= \left(1 - H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right)\right) \prod_{k \epsilon K_3} H\left(\Phi^k_{\Psi^k}(\mathbf{x})\right)^2, \\ \mathfrak{T}_3 &= \left(1 - H\left(\Phi^k_{\Psi^k}(\mathbf{x})\right)\right)^2 H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right) \prod_{\bar{k} \epsilon K_5} H\left(\Phi^{\bar{k}}_{\Psi^{\bar{k}}}(\mathbf{x})\right)^2, \\ \mathfrak{T}_4 &= H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right) \prod_{k \in K_1} H\left(\Phi^k_{\Psi^k}(\mathbf{x})\right)^2, \\ \mathfrak{T}_5 &= \left(1 - H\left(\Phi^n_{\Psi^n}(\mathbf{x})\right)\right) \mathfrak{T}_2.\end{aligned} \tag{4.36}$$

The terms associated with the step function in (4.35) have been squared so that it introduces extra terms in (4.36) as compared to (4.21). These extra terms alongwith the use of step function (Figure 4.6(b)), instead of the regularized version, prevents any evaluations of terms in (4.31) outside of defined object intensity regions. However, the delta function still needs to be regularized; otherwise, the boundary terms of (4.31) will not be evaluated unless the boundary happens to lie exactly on a pixel. Figure 4.8 demonstrates forces being computed with the modified model. The inward pulling force is identitcally zero outside the object region, while the outward pulling force is zero inside the object. The modified model produces correct results as shown in Figure 4.8(b), where all the forces become zero so that there is no more deformation of object shape. Note that the squaring of the terms in (4.35) does not change the minimum of the model.



(a) (b)

Figure 4.8: Evolution of the object contour in a scene with a single object using model terms (4.36) with step and regularized delta functions. (a) The initial frame, and (b) the target frame, show the evolution of object shape.

Chapter 5

Numerical Implementation

In this chapter, implementation details for the tracking model are given. This involves solving the rigid and nonrigid step of the combined tracking model, 4.1.3. The rigid part is solved using the Levenberg-Marquardt algorithm and the nonrigid part is solved using PDE discretization by finite difference method. Both the steps have a high computational complexity which are later addressed at the end.

5.1 Space and Time Discretization

In this chapter, the image domain is considered as two dimensional, i.e., $\Omega \in [0,1]^2$, with each point in the domain represented by $\mathbf{x} = (x, y)^\top$. So far, images have been considered as continuous functions defined over the image domain. In practice, the images are stored in digital format, so function values are available only on discrete points (pixels) in the domain, spaced at regular intervals from each other. Suppose the video frame and intensity template images consist of $M_x \times M_y$ pixels. The image domain $\Omega = [0,1]^2$ is discretized into a regular grid, with pixels located at (x_i, y_j)

$$\begin{cases} x_i = \dfrac{i}{h_x}, & \text{for } i = 1, 2, \ldots, M_x, \\ y_j = \dfrac{j}{h_y}, & \text{for } j = 1, 2, \ldots, M_y, \end{cases}$$

where the spacing between adjacent pixels in each dimension is given by

$$h_x = \frac{1}{M_x + 1}, \qquad\qquad h_y = \frac{1}{M_y + 1}.$$

The function values at the pixel locations are denoted by the following

$$\begin{aligned} F^t_{i,j} &= F(x_i, y_j, t), \\ f_{i,j} &= f(x_i, y_j), \qquad g_{i,j} = g(x_i, y_j). \end{aligned}$$

Other functions are discretized in a similar manner, with subscripts (i, j) indicating its discretized version.

The time dimension involved in the fluid model PDEs (4.30) and (4.32), is also discretized. Let $[0, \bar{T}]$ be the time interval for solving the PDE. This interval is discretized into L regular time steps. Time at each step is denoted by $\bar{t}_\ell = \ell \triangle \bar{t}$ for $\ell = 1, \ldots, L$ and $\triangle \bar{t} = \bar{T}/L$. The functions that evolve with this time are denoted by the superscript ℓ. For instance, the level set function for object n evolving in time are represented by

$$(\Phi^n_{\Psi^n})^\ell_{i,j} \equiv \Phi^n \left(\Psi^{n,\ell}(x_i, y_j) \right),$$

intensity templates by

$$(I^n_{\Psi^n})^\ell_{i,j} \equiv I^n \left(\Psi^{n,\ell}(x_i, y_j) \right),$$

and nonrigid transform by

$$(\Psi^n)^\ell_{i,j} \equiv (\hat{x}_i, \hat{y}_j) - \mathbf{r}^{n,\ell}(\hat{x}_i, \hat{y}_j),$$

where $(\hat{x}, \hat{y})^\top = \hat{\xi}^n(x, y) = Q_{\theta^n}(x, y)^\top + (\hat{\tau}^n_x, \hat{\tau}^n_y)^\top$ is the optimal transform from the rigid step.

5.2 Interpolation

The function values for images and level sets are available only at the pixel locations. The transforms acting on these functions may require function values be computed on a point that does not coincide with a pixel. Interpolation is required to compute these values. More precisely, let $(x, y) = \chi^n(x_i, y_j)$. In Figure 5.1, the point (x, y) lies in the middle of a region surrounded by four adjacent pixels (x_p, y_q), (x_{p+1}, y_q), (x_p, y_{q+1}),

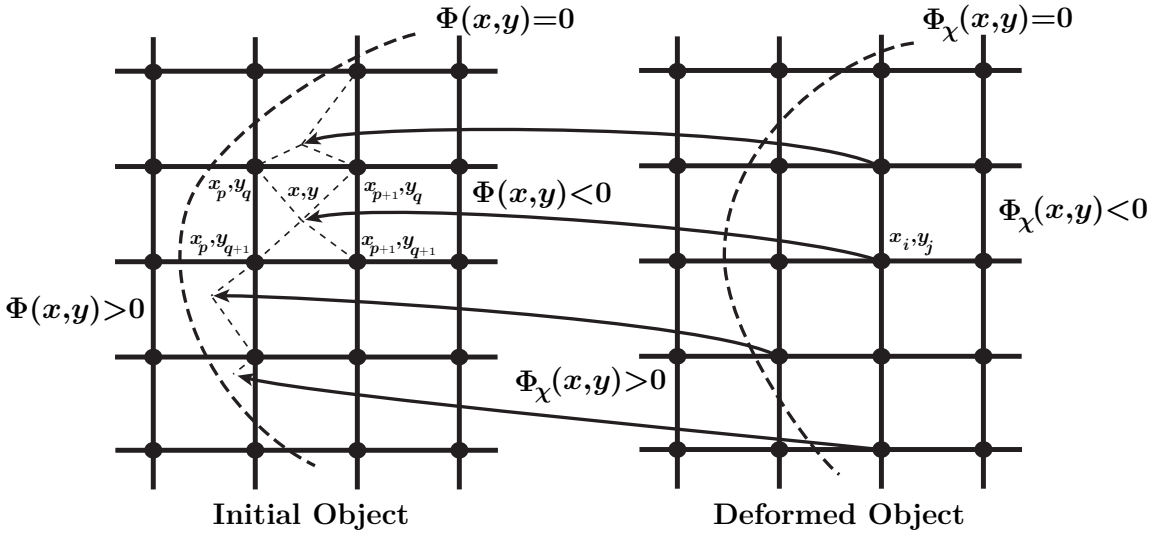


Figure 5.1: The figure shows pixels in the deformed object intensity image being interpolated from the pixels inside the initial object intensity image. The three types of interpolation displayed are interpolation between two, three, and four pixels from inside the object region in the initial object, and one case of a single pixel in which there is no interpolation.

and (x_{p+1}, y_{q+1}). Bilinear interpolation from these four pixels locations is normally performed to compute the function values:

$$\begin{aligned}(\Phi^n_{\chi^n})_{i,j} \approx & \frac{(x_{p+1}-x)}{h_x}\frac{(y_{p+1}-y)}{h_y}\Phi^n_{p,q} + \frac{(x_p-x)}{h_x}\frac{(y_{p+1}-y)}{h_y}\Phi^n_{p+1,q} \\ & + \frac{(x_{p+1}-x)}{h_x}\frac{(y-y_p)}{h_y}\Phi^n_{p,q+1} + \frac{(x-x_p)}{h_x}\frac{(y-y_p)}{h_y}\Phi^n_{p+1,q+1},\end{aligned} \tag{5.1}$$

$$\begin{aligned}(I^n_{\chi^n})_{i,j} \approx & \frac{(x_{p+1}-x)}{h_x}\frac{(y_{p+1}-y)}{h_y}I^n_{p,q} + \frac{(x_p-x)}{h_x}\frac{(y_{p+1}-y)}{h_y}I^n_{p+1,q} \\ & + \frac{(x_{p+1}-x)}{h_x}\frac{(y-y_p)}{h_y}I^n_{p,q+1} + \frac{(x-x_p)}{h_x}\frac{(y-y_p)}{h_y}I^n_{p+1,q+1}.\end{aligned} \tag{5.2}$$

The bilinear interpolation, though suitable for object interior and level set function, cannot be used for object intensity images near object boundaries. Some of the four pixels adjacent to (x, y) may lie outside the object region. Since the intensity is not

defined outside the object region (refer Section 4.5), (5.2) cannot be used. Interpolation is thus carried out using only the pixels lying inside the object. In case there is only one pixel inside the object, the value from that pixel is assigned. In the case of two pixels, if the two pixels lie along an axis, a line interpolates between the two pixels. The value at the required point is computed at the point lying closest to it on the line. If the two pixels are across from each other, then the nearest neighbor value is assigned to $(I^n_{\chi^n})_{i,j}$. For three pixels, a plane is defined and the required value is computed from the equation of a plane. One scenario of each of these cases is presented in Figure 5.1.

Interpolation techniques discussed here allow the images to be treated as continuous functions. Other techniques such as spline and wavelets may also be used, but the ones discussed here suffice. Both the rigid and nonrigid components of the tracking process use interpolation in the following sections.

5.3 Implementation: Rigid

In this section the implementation of the rigid tracking step is described. The model equation terms are first discretized. The Jacobian of the vector function is then derived. Later, the Levenberg-Marquardt algorithm is used to compute the optimal rigid parameters.

5.3.1 Discretization

The integrals in the reconstruction (4.22) and segmentation (4.26) measures are discretized in space with a rectangular quadrature,

$$\mathcal{D}^{rec} \approx \sum_{i=1}^{M_x} \sum_{j=1}^{M_y} \mathcal{D}^{rec}_{i,j}, \tag{5.3a}$$

$$\mathcal{D}^{seg} \approx \sum_{i=1}^{M_x} \sum_{j=1}^{M_y} \mathcal{D}^{seg}_{i,j}, \tag{5.3b}$$

where

$$\mathcal{D}^{rec}_{i,j} = \sum_{n=1}^{N} \left((\mathfrak{D}^n_1)_{i,j}\right)^2 + \left((\mathfrak{D}_2)_{i,j}\right)^2 \tag{5.4a}$$

$$\mathcal{D}_{i,j}^{seg} = \sum_{n=1}^{N} \left((\mathfrak{D}_3^n)_{i,j}\right)^2 + \left((\mathfrak{D}_4)_{i,j}\right)^2. \tag{5.4b}$$

The terms (4.23) and (4.27) in the model are discretized as

$$(\mathfrak{D}_1^n)_{i,j} \approx \left((I_{\xi^n}^n)_{i,j} - g_{i,j}\right)(\mathfrak{T}_5)_{i,j}\sqrt{h_x h_y}, \quad (\mathfrak{D}_2)_{i,j} \approx (B_{i,j} - g_{i,j})(\mathfrak{T}_6)_{i,j}\sqrt{h_x h_y}, \tag{5.5a}$$

$$(\mathfrak{D}_3^n)_{i,j} \approx (g_{i,j} - c_{\text{in}}^n)(\mathfrak{T}_5)_{i,j}\sqrt{h_x h_y}, \qquad (\mathfrak{D}_4)_{i,j} \approx (g_{i,j} - c_{\text{out}})(\mathfrak{T}_6)_{i,j}\sqrt{h_x h_y}. \tag{5.5b}$$

For the segmentation measure, the average inside and outside intensity is computed from the discretization of (4.15):

$$c_{\text{in}}^n \approx \frac{\sum_{i=1}^{M_x}\sum_{j=1}^{M_y} g_{i,j}(\mathfrak{T}_5)_{i,j} h_x h_y}{\sum_{i=1}^{M_x}\sum_{j=1}^{M_y}(\mathfrak{T}_5)_{i,j} h_x h_y}, \quad c_{\text{out}} \approx \frac{\sum_{i=1}^{M_x}\sum_{j=1}^{M_y} g_{i,j}(\mathfrak{T}_6)_{i,j} h_x h_y}{\sum_{i=1}^{M_x}\sum_{j=1}^{M_y}(\mathfrak{T}_6)_{i,j} h_x h_y}.$$

To compute the Jacobian of the similarity measure, the derivatives of (5.5a) and (5.5b) with respect to the rigid model parameters need to be derived, which are as follows:

$$\begin{aligned}\left(\frac{\partial \mathfrak{D}_1^n}{\partial s^n}\right)_{i,j} \approx{}& \left(-(\mathfrak{T}_1)_{i,j}\left((I_{\xi^n}^n)_{i,j} - g_{i,j}\right)(\mathfrak{T}_2)_{i,j}\right)\sqrt{h_x h_y} \\ &+ \left(\left(\nabla\nabla I_{\xi^n}^n\right)_{i,j}\frac{\partial \xi^n}{\partial s^n}(\mathfrak{T}_5)_{i,j}\right)\sqrt{h_x h_y},\end{aligned} \tag{5.6a}$$

$$\left(\frac{\partial \mathfrak{D}_2}{\partial s^n}\right)_{i,j} \approx (\mathfrak{T}_1)_{i,j}(B_{i,j} - g_{i,j})(\mathfrak{T}_4)_{i,j}, \tag{5.6b}$$

$$\left(\frac{\partial \mathfrak{D}_3^n}{\partial s^n}\right)_{i,j} \approx \left(-(\mathfrak{T}_1)_{i,j}(g_{i,j} - c_{\text{in}}^n)(\mathfrak{T}_2)_{i,j} - \frac{\partial c_{\text{in}}^n}{\partial s^n}(\mathfrak{T}_5)_{i,j}\right)\sqrt{h_x h_y}, \tag{5.6c}$$

$$\left(\frac{\partial \mathfrak{D}_4}{\partial s^n}\right)_{i,j} \approx \left((\mathfrak{T}_1)_{i,j}(g_{i,j} - c_{\text{out}})(\mathfrak{T}_4)_{i,j} - \frac{\partial c_{\text{out}}}{\partial s^n}(\mathfrak{T}_6)_{i,j}\right)\sqrt{h_x h_y}, \tag{5.6d}$$

where $\partial \xi^n / \partial s^n$ is given by (4.25) and the derivatives of average inside and outside intensities are

$$\begin{aligned}\frac{\partial c_{\text{in}}^n}{\partial s^n} \approx{}& -\frac{\sum_{i=1,j=1}^{M_x,M_y} g_{i,j}(\mathfrak{T}_1)_{i,j}(\mathfrak{T}_2)_{i,j}}{\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_5)_{i,j} h_x h_y} \\ &+ \frac{\left(\sum_{i=1,j=1}^{M_x,M_y} g_{i,j}(\mathfrak{T}_5)_{i,j}\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_1)_{i,j}(\mathfrak{T}_2)_{i,j}\right)(h_x h_y)^2}{\left(\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_5)_{i,j} h_x h_y\right)^2},\end{aligned} \tag{5.7a}$$

$$\frac{\partial c_{\text{in}}^k}{\partial s^n} \approx \frac{\sum_{i=1,j=1}^{M_x,M_y} g_{i,j}(\mathfrak{T}_1)_{i,j}(\mathfrak{T}_3)_{i,j}h_xh_y}{\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_7)_{i,j}h_xh_y} - \frac{\left(\sum_{i=1,j=1}^{M_x,M_y} g_{i,j}(\mathfrak{T}_7)_{i,j}\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_1)_{i,j}(\mathfrak{T}_3)_{i,j}\right)(h_xh_y)^2}{\left(\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_7)_{i,j}h_xh_y\right)^2}, \tag{5.7b}$$

$$\frac{\partial c_{\text{out}}}{\partial s^n} \approx \frac{\sum_{i=1,j=1}^{M_x,M_y} g_{i,j}(\mathfrak{T}_1)_{i,j}(\mathfrak{T}_4)_{i,j}h_xh_y}{\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_6)_{i,j}h_xh_y} - \frac{\left(\sum_{i=1,j=1}^{M_x,M_y} g_{i,j}(\mathfrak{T}_6)_{i,j}\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_1)_{i,j}(\mathfrak{T}_4)_{i,j}\right)(h_xh_y)^2}{\left(\sum_{i=1,j=1}^{M_x,M_y}(\mathfrak{T}_6)_{i,j}h_xh_y\right)^2}. \tag{5.7c}$$

The gradient of the object intensity image and the level set function, present in (5.13a) and $\mathfrak{T}_1$ respectively, are approximated by central finite differences:

$$\left(\nabla I_{\xi^n}^n\right)_{i,j} \approx \begin{bmatrix} \dfrac{(I_{\xi^n}^n)_{i+1,j} - 2(I_{\xi^n}^n)_{i,j} + (I_{\xi^n}^n)_{i-1,j}}{h_x^2} \\ \dfrac{(I_{\xi^n}^n)_{i,j+1} - 2(I_{\xi^n}^n)_{i,j} + (I_{\xi^n}^n)_{i,j-1}}{h_y^2} \end{bmatrix}, \tag{5.8a}$$

$$\left(\nabla \Phi_{\xi^n}^n\right)_{i,j} \approx \begin{bmatrix} \dfrac{(\Phi_{\xi^n}^n)_{i+1,j} - 2(\Phi_{\xi^n}^n)_{i,j} + (\Phi_{\xi^n}^n)_{i-1,j}}{h_x^2} \\ \dfrac{(\Phi_{\xi^n}^n)_{i,j+1} - 2(\Phi_{\xi^n}^n)_{i,j} + (\Phi_{\xi^n}^n)_{i,j-1}}{h_y^2} \end{bmatrix}. \tag{5.8b}$$

In the implementation, the object is not rotated in the rigid step. Since most of the object's rigid motion arises from translation in many videos sequences, rotation can be ignored in these instances. Small degrees of rotation is still captured by the nonrigid step so that tracking works accurately. The rotation matrix is set to be the identity matrix, i.e., $Q_{\theta^n} = \mathbf{I}_2$, so that $\xi^n(x,y) = (x,y)^\top + (\tau_x^n, \tau_y^n)^\top$.

5.3.2 Algorithm

The rigid step of the model is solved using Levenberg-Marquardt algorithm [22] which is based on the Gauss-Newton [22] and gradient descent method. The algorithm requires

the similarity measure (5.3a) and (5.3b) be expressed as sum of the squares. The similarity measure is expressed in terms of its components, and assembled into a vector $\mathbb{R}^{(N+1)M_xM_y}$ given by

$$\begin{aligned}\vec{\mathcal{D}}^{rec}\left[\vec{s}\right] &= \begin{pmatrix} (\bar{\mathfrak{D}}_1^1)_m & \cdots & (\bar{\mathfrak{D}}_1^N)_m & (\bar{\mathfrak{D}}_2)_m \end{pmatrix}^\top, \\ \vec{\mathcal{D}}^{seg}\left[\vec{s}\right] &= \begin{pmatrix} (\bar{\mathfrak{D}}_3^1)_m & \cdots & (\bar{\mathfrak{D}}_3^N)_m & (\bar{\mathfrak{D}}_4)_m \end{pmatrix}^\top, \end{aligned} \tag{5.9}$$

where

$$\begin{cases} (\bar{\mathfrak{D}}_1^n)_m = (\mathfrak{D}_1^n)_{i,j}, \quad (\bar{\mathfrak{D}}_2)_m = (\mathfrak{D}_2)_{i,j}, & m = j + M_y(i-1) \\ & \substack{1\le i\le M_x \\ 1\le j\le M_y} \\ (\bar{\mathfrak{D}}_3^n)_m = (\mathfrak{D}_3^n)_{i,j}, \quad (\bar{\mathfrak{D}}_4)_m = (\mathfrak{D}_4)_{i,j}. & \end{cases}$$

The parameters are also assembled into a vector $\mathbb{R}^{2N}$

$$\vec{s} = \left[\tau_x^1, \tau_y^1, \tau_x^2, \tau_y^2, \ldots, \tau_x^N, \tau_y^N\right]^\top.$$

It also requires computing the Jacobian of the similarity measures vector (5.9), which is a matrix $\mathbb{R}^{(N+1)M_xM_y\times 2N}$. The columns of the Jacobian matrix are as follows:

$$\begin{aligned}\mathcal{J}^{rec}\left[\vec{s}\right] &= \left[\frac{\partial\vec{\mathcal{D}}^{rec}}{\partial\tau^1}, \frac{\partial\vec{\mathcal{D}}^{rec}}{\partial\tau^2}, \ldots, \frac{\partial\vec{\mathcal{D}}^{rec}}{\partial\tau^N}\right], \\ \mathcal{J}^{seg}\left[\vec{s}\right] &= \left[\frac{\partial\vec{\mathcal{D}}^{seg}}{\partial\tau^1}, \frac{\partial\vec{\mathcal{D}}^{seg}}{\partial\tau^2}, \ldots, \frac{\partial\vec{\mathcal{D}}^{seg}}{\partial\tau^N}\right]. \end{aligned} \tag{5.10}$$

where

$$\frac{\partial\vec{\mathcal{D}}^{rec}}{\partial\tau^n} = \begin{bmatrix} \left(\frac{\partial\bar{\mathfrak{D}}_1^1}{\partial\tau_x^n}\right)_m & \left(\frac{\partial\bar{\mathfrak{D}}_1^1}{\partial\tau_y^n}\right)_m \\ \vdots & \\ \left(\frac{\partial\bar{\mathfrak{D}}_1^N}{\partial\tau_x^n}\right)_m & \left(\frac{\partial\bar{\mathfrak{D}}_1^N}{\partial\tau_y^n}\right)_m \\ \left(\frac{\partial\bar{\mathfrak{D}}_2}{\partial\tau_x^n}\right)_m & \left(\frac{\partial\bar{\mathfrak{D}}_2}{\partial\tau_y^n}\right)_m \end{bmatrix}, \qquad \frac{\partial\vec{\mathcal{D}}^{seg}}{\partial\tau^n} = \begin{bmatrix} \left(\frac{\partial\bar{\mathfrak{D}}_3^1}{\partial\tau_x^n}\right)_m & \left(\frac{\partial\bar{\mathfrak{D}}_3^1}{\partial\tau_y^n}\right)_m \\ \vdots & \\ \left(\frac{\partial\bar{\mathfrak{D}}_3^N}{\partial\tau_x^n}\right)_m & \left(\frac{\partial\bar{\mathfrak{D}}_3^N}{\partial\tau_y^n}\right)_m \\ \left(\frac{\partial\bar{\mathfrak{D}}_4}{\partial\tau_x^n}\right)_m & \left(\frac{\partial\bar{\mathfrak{D}}_4}{\partial\tau_y^n}\right)_m \end{bmatrix}.$$

The derivatives are given by

$$\begin{cases} \left(\frac{\partial\bar{\mathfrak{D}}_1^{\bar{n}}}{\partial s^n}\right)_m = \left(\frac{\partial\mathfrak{D}_1^{\bar{n}}}{\partial s^n}\right)_{i,j}, \quad \left(\frac{\partial\bar{\mathfrak{D}}_2}{\partial s^n}\right)_m = \left(\frac{\partial\mathfrak{D}_2}{\partial s^n}\right)_{i,j}, & \bar{n} = 1, \ldots, N \\ & m = j + M_y(i-1) \\ \left(\frac{\partial\bar{\mathfrak{D}}_4^{\bar{n}}}{\partial s^n}\right)_m = \left(\frac{\partial\mathfrak{D}_3^{\bar{n}}}{\partial s^n}\right)_{i,j}, \quad \left(\frac{\partial\bar{\mathfrak{D}}_4}{\partial s^n}\right)_m = \left(\frac{\partial\mathfrak{D}_4}{\partial s^n}\right)_{i,j}, & \substack{1\le i\le M_x \\ 1\le j\le M_y} \end{cases}$$

with the model parameters $s^n = \{\tau_x^n, \tau_y^n\}$.

The Levenberg-Marquardt algorithm starts with an initial estimate $\vec{s}_0$ of the parameter vector and computes updates to it by the following relation

$$\vec{s}_{\ell+1} = \vec{s}_\ell + \left(\mathcal{J}^\top \mathcal{J} + \eta_\ell \text{diag}(\mathcal{J}^\top \mathcal{J})\right)^{-1} \mathcal{J}^\top \vec{\mathcal{D}}.$$

Here, $\vec{\mathcal{D}} = \vec{\mathcal{D}}^{rec}$ or $\vec{\mathcal{D}}^{seg}$ and $\mathcal{J}$ is the Jacobian matrix from (5.10). The parameter $\eta_\ell \in \mathbb{R}^+$ is adjusted in each iteration to obtain gradient descent like behavior when $\vec{s}_\ell$ is far from the minimum and Gauss-Newton like behavior when it is close. The details of the procedure are given in Algorithm 5.1.

Algorithm 5.1 Levenberg-Marquardt for Rigid Tracking

Input: Object information: $\Phi^n(x_i, y_j), I^n(x_i, y_j)$ for $n = 1, 2, \ldots, N$,
Initial parameters: $\vec{s}_0, \eta_0$,
Trust region parameters: $0 < \bar{\omega}_{\text{down}} < 1 < \bar{\omega}_{\text{up}}$, and $\bar{\mu}_0 \leq \bar{\mu}_{\text{low}} < \bar{\mu}_{\text{high}}$.

Output: Optimal parameters: $\hat{s}$.

Compute residual vector $\vec{\mathcal{D}}[\vec{s}_\ell]$ *from* (5.9)*, and Jacobian* $\mathcal{J} := \mathcal{J}[\vec{s}_\ell]$ *from* (5.10)*.*

$\ell \leftarrow 0$, TERMINATE $\leftarrow$ **false**
repeat
  $\vec{s}_{\ell+1} \leftarrow \left(\mathcal{J}^\top \mathcal{J} + \eta_\ell \text{diag}(\mathcal{J}^\top \mathcal{J})\right)^{-1} \mathcal{J}^\top \vec{\mathcal{D}}[\vec{s}_\ell]$
  $\delta s \leftarrow \vec{s}_{\ell+1} - \vec{s}_\ell$
  if $\ell = \ell_{max}$ or $\left\|\mathcal{J}^\top \vec{\mathcal{D}}[\vec{s}_\ell]\right\|_\infty \leq \varepsilon_1$ or $\delta s / \vec{s}_\ell \leq \varepsilon_2$ or $\left\|\vec{\mathcal{D}}[\vec{s}_\ell]\right\|_2 \leq \epsilon_3$ **then**
    TERMINATE $\leftarrow$ **true**
  else
    $\bar{\rho} \leftarrow 0$
    while $\bar{\rho} < \bar{\mu}_0$ **do**
      $\bar{\rho} \leftarrow \left(\left\|\vec{\mathcal{D}}[\vec{s}_{\ell+1}]\right\|_2^2 - \left\|\vec{\mathcal{D}}[\vec{s}_\ell]\right\|_2^2\right) / \left(\delta s^\top \mathcal{J}^\top \vec{\mathcal{D}}[\vec{s}_\ell]\right)$
      $\eta_{\ell+1} \leftarrow \max(\bar{\omega}_{\text{up}} \eta_\ell, \eta_0)$
      $\vec{s}_{\ell+1} \leftarrow \left(\mathcal{J}^\top \mathcal{J} + \eta_\ell \text{diag}(\mathcal{J}^\top \mathcal{J})\right)^{-1} \mathcal{J}^\top \vec{\mathcal{D}}[\vec{s}_\ell]$
      $\delta s \leftarrow \vec{s}_{\ell+1} - \vec{s}_\ell$
    end while
    if $\bar{\mu}_0 \leq \bar{\rho} < \bar{\mu}_{\text{low}}$ **then**
      $\eta_{\ell+1} \leftarrow \max(\bar{\omega}_{\text{up}} \eta_\ell, \eta_0)$
    else if $\bar{\rho} > \bar{\mu}_{\text{high}}$ **then**
      $\eta_{\ell+1} \leftarrow \bar{\omega}_{\text{down}} \eta_\ell$
    end if
    if $\eta_{\ell+1} < \eta_0$ **then**
      $\eta_{\ell+1} \leftarrow 0$
    end if
    $\ell \leftarrow \ell + 1$
  end if
until TERMINATE $\neq$ **true**
$\hat{s} \leftarrow \vec{s}_\ell$

5.4 Implementation: Nonrigid

This section discusses the implementation of the nonrigid part of the tracking model. The nonrigid tracking equations are discretized and a method is provided to compute the solution. The vectors defined on the domain therefore have components along the horizontal and vertical axis. The components of the vectors are denoted by $\mathbf{v} = (v_x, v_y)^\top$, $\mathbf{r} = (r_x, r_y)^\top$ and $\mathbf{b} = (b_x, b_y)^\top$.

5.4.1 Discretization

The Navier Lamé equations in (4.30) and (4.32), in two dimensions are written as:

$$\begin{aligned} A[\mathbf{v}^n] &= \mu \triangle \mathbf{v}^n + (\lambda + \mu) \nabla (\nabla . \mathbf{v}^n) \\ &= \begin{pmatrix} (\lambda + 2\mu) \partial_{xx} v_x^n + \mu \partial_{yy} v_x^n + (\lambda + \mu) \partial_{xy} v_y^n \\ (\lambda + \mu) \partial_{xy} v_x^n + \mu \partial_{xx} v_y^n + (\lambda + 2\mu) \partial_{yy} v_y^n \end{pmatrix}. \end{aligned} \tag{5.11}$$

The derivatives in these equations are discretized and approximated by second order central finite differences

$$\begin{aligned} \left(\frac{\partial^2 v_x^n}{\partial x^2}\right)_{i,j}^{\ell} &\approx \frac{(v_x^n)_{i+1,j}^{\ell} - 2(v_x^n)_{i,j}^{\ell} + (v_x^n)_{i-1,j}^{\ell}}{h_x^2}, \\ \left(\frac{\partial^2 v_x^n}{\partial y^2}\right)_{i,j}^{\ell} &\approx \frac{(v_x^n)_{i,j+1}^{\ell} - 2(v_x^n)_{i,j}^{\ell} + (v_x^n)_{i,j-1}^{\ell}}{h_y^2}, \\ \left(\frac{\partial^2 v_x^n}{\partial x \partial y}\right)_{i,j}^{\ell} &\approx \frac{(v_x^n)_{i+1,j+1}^{\ell} - (v_x^n)_{i+1,j-1}^{\ell} - (v_x^n)_{i-1,j+1}^{\ell} + (v_x^n)_{i-1,j-1}^{\ell}}{4h_x h_y}, \\ \left(\frac{\partial^2 v_y^n}{\partial x^2}\right)_{i,j}^{\ell} &\approx \frac{(v_y^n)_{i+1,j}^{\ell} - 2(v_y^n)_{i,j}^{\ell} + (v_y^n)_{i-1,j}^{\ell}}{h_x^2}, \\ \left(\frac{\partial^2 v_y^n}{\partial y^2}\right)_{i,j}^{\ell} &\approx \frac{(v_y^n)_{i,j+1}^{\ell} - 2(v_y^n)_{i,j}^{\ell} + (v_y^n)_{i,j-1}^{\ell}}{h_y^2}, \\ \left(\frac{\partial^2 v_y^n}{\partial x \partial y}\right)_{i,j}^{\ell} &\approx \frac{(v_y^n)_{i+1,j+1}^{\ell} - (v_y^n)_{i+1,j-1}^{\ell} - (v_y^n)_{i-1,j+1}^{\ell} + (v_y^n)_{i-1,j-1}^{\ell}}{4h_x h_y}. \end{aligned}$$

The body force in (4.30) for reconstruction based models is given by

$$(b_x^n)_{i,j}^{\ell} = \left(\frac{\partial \mathcal{D}^{rec}}{\partial r_x^n}\right)_{i,j}^{\ell}, \quad (b_y^n)_{i,j}^{\ell} = \left(\frac{\partial \mathcal{D}^{rec}}{\partial r_y^n}\right)_{i,j}^{\ell}, \tag{5.12a}$$

and body force in (4.32) for the segmentation based model is given by

$$(b_x^n)_{i,j}^\ell = \left(\frac{\partial \mathcal{D}^{seg}}{\partial r_x^n}\right)_{i,j}^\ell, \quad (b_y^n)_{i,j}^\ell = \left(\frac{\partial \mathcal{D}^{seg}}{\partial r_y^n}\right)_{i,j}^\ell. \tag{5.12b}$$

The variation of the similarity measures with respect to the displacement field $\mathbf{r}^n = (r_x^n, r_y^n)$, are also discretized to give

$$\begin{aligned}\left(\frac{\partial \mathcal{D}^{rec}}{\partial \mathbf{r}^n}\right)_{i,j}^\ell = \; & (\mathfrak{T}_1)_{i,j}^\ell \Bigg(- \left((I_{\Psi^n}^n)_{i,j}^\ell - g_{i,j}\right)^2 (\mathfrak{T}_2)_{i,j}^\ell \\ & + \sum_{k \epsilon K_2} \left((I_{\Psi^k}^k)_{i,j}^\ell - g_{i,j}\right)^2 (\mathfrak{T}_3)_{i,j}^\ell + (B_{i,j} - g_{i,j})^2 (\mathfrak{T}_4)_{i,j}^\ell \Bigg) \\ & + 2\left((I_{\Psi^n}^n)_{i,j}^\ell - g_{i,j}\right)(-1)(\mathfrak{T}_5)_{i,j}^\ell,\end{aligned} \tag{5.13a}$$

$$\begin{aligned}\left(\frac{\partial \mathcal{D}^{seg}}{\partial \mathbf{r}^n}\right)_{i,j}^\ell = \; & (\mathfrak{T}_1)_{i,j}^\ell \Bigg(- \left(g_{i,j} - c_{\text{in}}^n\right)^2 (\mathfrak{T}_2)_{i,j}^\ell + \sum_{k \epsilon K_2} \left(g_{i,j} - c_{\text{in}}^k\right)^2 (\mathfrak{T}_3)_{i,j}^\ell \\ & + \left(g_{i,j} - c_{\text{out}}\right)^2 (\mathfrak{T}_4)_{i,j}^\ell \Bigg) - 2\left(g(x,y) - c_{\text{in}}^n\right)^2 \frac{\partial c_{\text{in}}^n}{\partial \mathbf{r}^n} (\mathfrak{T}_5)_{i,j}^\ell \\ & - \sum_{k \epsilon K_2} 2\left(g_{i,j} - c_{\text{in}}^k\right) \frac{\partial c_{\text{in}}^k}{\partial \mathbf{r}^n} (\mathfrak{T}_7)_{i,j}^\ell - 2\left(g_{i,j} - c_{\text{out}}\right) \frac{\partial c_{\text{out}}}{\partial \mathbf{r}^n} (\mathfrak{T}_6)_{i,j}^\ell.\end{aligned} \tag{5.13b}$$

The terms $(\mathfrak{T}_m)_{i,j}^\ell = \mathfrak{T}_m^\ell(x_i, y_j)$ for $m = 1, \ldots, 7$ as given in (4.36) for the reconstruction model and in (4.21) for the segmentation model, are discretized in a similar manner. The discretizations of $\partial c_{\text{in}}^n / \partial \mathbf{r}^n$, $\partial c_{\text{in}}^k / \partial \mathbf{r}^n$, and $\partial c_{\text{out}} / \partial \mathbf{r}^n$ are as defined in (5.7a), (5.7b), and (5.7c). The gradient of the intensity templates, $\nabla I_{\Psi^n}^n(x,y)$, and the level set, $\nabla \Phi_{\Psi^n}^n(x,y)$, are discretized as in (5.8a) and (5.8b) respectively, but replacing $\xi^n(x,y)$ with $\Psi^{n,\ell}(x,y)$. The equation to update the velocity for each object is

$$\left(A\left[(v_x^n, v_y^n)^\top\right]\right)_{i,j} = \left((b_x^n)_{i,j}^\ell, (b_y^n)_{i,j}^\ell\right)^\top. \tag{5.14}$$

The time derivative in the second equation of the pairs in (4.30) and (4.32), is discretized with the forward Euler method:

$$\left(\frac{\partial r_x^n}{\partial \bar{t}}\right)_{i,j}^\ell = \frac{(r_x^n)_{i,j}^{\ell+1} - (r_x^n)_{i,j}^\ell}{\triangle \bar{t}},$$
$$\left(\frac{\partial r_y^n}{\partial \bar{t}}\right)_{i,j}^\ell = \frac{(r_y^n)_{i,j}^{\ell+1} - (r_y^n)_{i,j}^\ell}{\triangle \bar{t}}.$$

The update equation for the displacement is obtained by rearranging the terms in (4.30) and (4.32):

$$\begin{aligned}(r_x^n)_{i,j}^{\ell+1} &= (v_x^n)_{i,j}^{\ell}\left(1-\frac{(r_x^n)_{i+1,j}^{\ell}-(r_x^n)_{i-1,j}^{\ell}}{2\triangle\bar{t}}\right),\\ (r_y^n)_{i,j}^{\ell+1} &= (v_y^n)_{i,j}^{\ell}\left(1-\frac{(r_y^n)_{i,j+1}^{\ell}-(r_y^n)_{i,j-1}^{\ell}}{2\triangle\bar{t}}\right).\end{aligned} \tag{5.15}$$

The Dirichlet boundary conditions are enforced for the velocity field on the boundary of the image domain. These are stated by:

$$\begin{cases}(r_x^n)_{i,0}^{\ell}=0, \quad (r_x^n)_{i,M_y+1}^{\ell}=0, \quad (r_x^n)_{0,j}^{\ell}=0, \quad (r_x^n)_{M_x+1,j}^{\ell}=0, & i=0,\ldots,M_x+1\\ (r_y^n)_{i,0}^{\ell}=0, \quad (r_y^n)_{i,M_y+1}^{\ell}=0, \quad (r_y^n)_{0,j}^{\ell}=0, \quad (r_y^n)_{M_x+1,j}^{\ell}=0. & j=0,\ldots,M_y+1\end{cases}$$

The zero initial conditions are specified for velocity and displacements:

$$\begin{cases}(r_x^n)_{i,j}^{0}=0, \quad (r_y^n)_{i,j}^{0}=0, & i=0,\ldots,M_x+1\\ (v_x^n)_{i,j}^{0}=0, \quad (v_y^n)_{i,j}^{0}=0. & j=0,\ldots,M_y+1\end{cases}$$

5.4.2 Iteration Process

The system of equations in (4.30) and (4.32) are solved simultaneously for computing the nonrigid deformation for all objects. Each pair of equations requires two steps, one for computing the velocity generated by the applied body force, and the other for computing the displacement from the velocity. The body force equations introduce coupling into the pair of equations since velocity is computed from the force, which is dependent on displacement, and the displacement is computed from the velocity. The two equations are solved iteratively until the approximate solution converges to the required solution. We use $\left\|\mathbf{v}^{n,\ell}\right\|_{\infty} < tol$ and maximum number of iterations as the stopping criterion. At the same time iteration is also performed on the system of PDEs to solve them simultaneously. A brief outline of the steps is provided in Algorithm 5.2.

The reconstruction based measure in (4.11) will usually be nonzero even if the computed deformation is a global optimum. The reason is that in real world video sequences, the object intensity may change due to changes in illumination, shadows, and object pose. It may thus not be possible to reconstruct the target frame exactly from the information in the previous frame. The operational assumption is that the changes occur gradually from frame to frame so that they are easily captured by the tracking process.

Algorithm 5.2 Nonrigid tracking

Input: Object information: $\Phi^n(x_i, y_j), I^n(x_i, y_j)$, for $n = 1, 2, \ldots, N$
Optimal rigid parameters: $\hat{\boldsymbol{\tau}}$

Output: Optimal nonrigid parameters: $\vec{\mathbf{r}}(x_i, y_j)$

$r_x^{n,0} \leftarrow 0$, $r_y^{n,0} \leftarrow 0$, $v_x^{n,0} \leftarrow 0$, $v_y^{n,0} \leftarrow 0$, for $n = 1, 2, \ldots, N$.
CONVERGED^n ←**false**, for $n = 1, 2, \ldots, N$.
for $\ell = 0, 1, \ldots, L-1$ **do**
  for all n such that $n = 1, 2, \ldots, N$ and CONVERGED^n =**false do**
    Compute $(v_x^n)_{i,j}^{\ell+1}$ and $(v_y^n)_{i,j}^{\ell+1}$ from (5.14)
    Compute $(r_x^n)_{i,j}^{\ell+1}$ and $(r_y^n)_{i,j}^{\ell+1}$ from (5.15)
    if $\left\|\mathbf{v}^{n,\ell+1}\right\|_\infty < tol$ **then**
      CONVERGED^n ←**true**
    end if
    Compute Jacobian $\bar{\mathcal{J}}^{n,\ell}$ of the transformation from (5.16)
    if $\min_{i,j} \left|\bar{\mathcal{J}}^{n,\ell}(x_i, y_j)\right| < tol_{\bar{\mathcal{J}}}$ **then**
      Regrid for object n
    end if
  end for
end for

5.4.2.1 Regridding

The fluid PDE requires regridding when the deformation becomes singular. The Jacobian of the deformation

$$\bar{\mathcal{J}}^{n,\ell}(x, y) = \mathbf{I}_d - \nabla \mathbf{r}^{n,\ell}(x, y) \tag{5.16}$$

is computed for each grid point, for a given object. If the determinant of the Jacobian $\left|\bar{\mathcal{J}}^{n,\ell}(x, y)\right| = 0$ for any $(x, y) \in \Omega$, the transformation map is no longer bijective at that point. To prevent singularities, the displacement field is regridded if the determinant falls below a certain threshold, i.e., $\min_{i,j} \left|\bar{\mathcal{J}}^{n,\ell}(x_i, y_j)\right| < tol_{\bar{\mathcal{J}}}$. The regridding process involves forming an intermediate set of deformed intensity templates and level set functions. The velocity and displacement are then set to zero and the PDE is solved again with the deformed objects as the initial object.

5.5 Algorithm

In this section we give an outline of the tracking process in Algorithm 5.3. We also discuss performance issues that arise from the computational complexity of the process, and ways to mitigate these issues.

Step 1 of the algorithm requires the objects be initially segmented using any contour segmentation algorithm. The segmentation provides the level set functions for the objects which are used in the tracking. It is also assumed that all the objects are unoccluded in the first frame so that the intensity templates for the whole object is extracted. The tracking process is then started for all frames in the video. First the rigid tracking step 2 is performed, which gives the estimated rigid motion parameters as well as the depth order. After that the nonrigid tracking step is performed 3, while keeping the depth order fixed. Once all the rigid and nonrigid tracking parameters are obtained, the updated level sets and intensity templates are computed. The same procedure is repeated for subsequent frames starting with the updated information.

5.5.1 Recovery of Occluded Region

During the process of tracking in a given initial frame, if all the objects are unoccluded then the intensity information of the objects can be obtained directly from the initial frame. However, if in a given frame objects are occluded, then the intensity information for the occluded regions cannot be obtained from the image frame. For the reconstruction based model, the intensity of the whole object is required in order to form the reconstructed image. Without this piece of information tracking correctly into the next frame is not possible. Our approach to recover the missing information is to use the object's intensity from the last unoccluded frame. The steps for this are given in 4 of the algorithm. The intensity templates are updated only in the unoccluded region of the object. The intensity information in the occluded region is retained until that region becomes unoccluded. This way new information of the object is updated as it becomes available, while the past information is used otherwise. Tracking can then be performed through several frames of occlusion. Note, however, if the object remains occluded for a long period of time, the intensity information may no longer be accurate due to changes in the occluded region.

Algorithm 5.3 Tracking algorithm

1. Segment objects in the first frame $F(x, y, 1)$. For each object $n = 1, \ldots, N$ compute the level set functions for representing the contour

$$\Phi^n(x, y) \leq 0, \ \mathbf{x} \text{ inside object } n,$$

and intensity image

$$I^n(x, y) = \begin{cases} F(x, y, 1) & \Phi^n(x, y) \leq 0, \\ 0 & \text{otherwise.} \end{cases}$$

2. Rigid tracking. Solve the optimization problem (4.4) or (4.14) as outlined in Algorithm 5.1. Denote the optimal parameters as $\hat{\boldsymbol{\tau}}$ and $\hat{z}$.

3. Nonrigid tracking. Solve the Euler-Lagrange equations for nonrigid motion $\vec{\mathbf{r}}$ (4.32) or (4.32) as outlined in Algorithm 5.2. Use the parameters obtained from the rigid tracking step and the transform $\Psi^n(\mathbf{x}) = (\hat{x}, \hat{y}) + \mathbf{r}^n((\hat{x}, \hat{y}))$, where $(\hat{x}, \hat{y}) = (x, y)^\top + (\hat{\tau}_x^n, \hat{\tau}_y^n)^\top$, in these equations.

4. Update object information for the next frame. For $n = 1, \ldots, N$

$$\Phi^n(x, y) = \Phi^n_{\Psi^n}(x, y).$$

If reconstruction based measure is used also update

$$I^n(x, y) = \begin{cases} F(x, y, t+1) & \Phi^n_{\Psi^n}(x, y) \leq 0 \text{ and } \Phi^{\hat{z}(k)}_{\Psi^{\hat{z}(k)}}(x, y) > 0 \text{ for } k \in K_3, \\ I^n_{\Psi^n}(x, y) & \Phi^n_{\Psi^n}(x, y) \leq 0 \text{ and for any } \Phi^{\hat{z}(k)}_{\Psi^{\hat{z}(k)}}(x, y) \leq 0 \text{ for } k \in K_3, \\ R(x, z, \mathbf{u}) & \text{otherwise.} \end{cases}$$

5. Reinitialize level sets

6. Repeat Step 2 to 5 for all frames $t = 1, \ldots, t_{max}$.

5.5.2 Level Set Reinitialization

The level set functions are initially set to the signed distance function. This is to ensure that the gradient of the level set used in the computation of $\mathfrak{T}_1$ is uniform along the boundary and regions around it. The shape of the level set function, after tracking of each frame, no longer remains a signed distance function. This may cause numerical issues and eventually introduce a nonsmooth zero level contour. The level set function is therefore reinitialized after every frame. One method to reinitialize the level sets is to solve the PDE

$$\frac{\partial \Phi^n(x,y)}{\partial \bar{t}} = 1 - \left|\nabla \Phi^n(x,y)\right|,$$

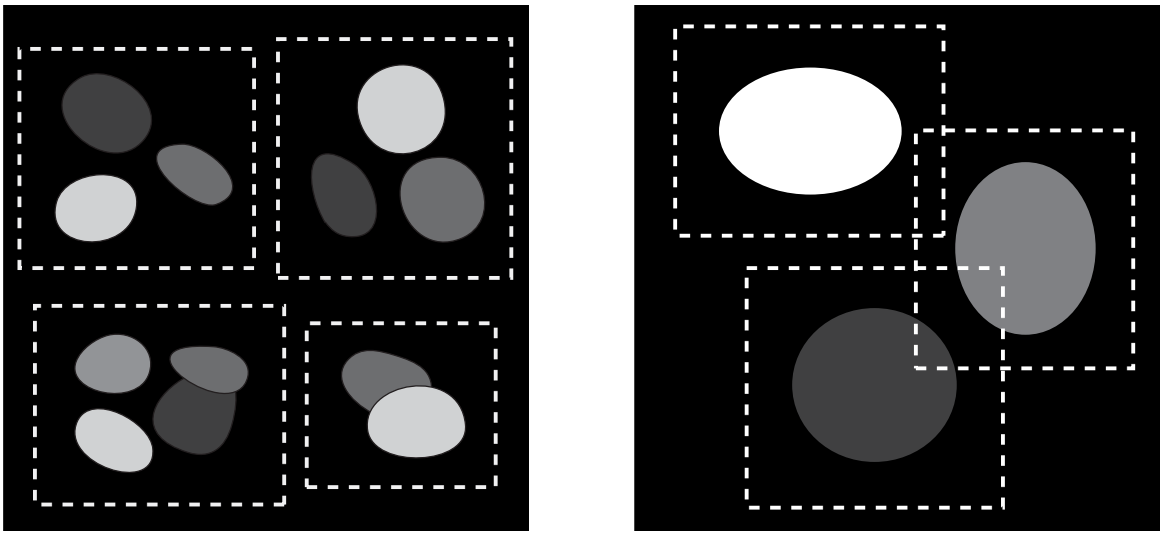
see [42] for details. The steady state of this PDE produces a signed distance function.

5.5.3 Performance Issues

We note that the number of depth order parameters in step 2 increases in the order of $O(n!)$ with increasing number of objects. For instance, tracking for just four objects requires 24 different depth orders and additionally 8 translation parameters. In practice, the objects move continuously from frame to frame in the vicinity of its past frame location. Far away objects will not occlude each other. Thus, many of the depth order permutations can be ignored.

It is assumed that objects do not undergo rapid motion between two frames. Suppose an object may move at most 10 pixels in a direction along the x or y axis. Let a window be constructed around each object in the initial frame by adding a fixed length of 10 pixels along each face of its minimum bounding rectangle, which is the smallest rectangle that fits the object. The window formalizes the notion of maximum motion of an object from the initial frame to the target frame, as shown in Figure 5.2(b).

First, clusters of objects are detected in the scene. A cluster is a set of objects that lie close together. There may be several such clusters, as shown in Figure 5.2(a), but none of them share any objects between themselves, since the objects in different clusters are far from each other. A window over the cluster is defined by forming a minimum bounding rectangle over all of the objects in the cluster set, and adding the fixed length of 10 pixels to it. The cluster window is larger than or equal to any of the individual object windows inside it. Also, no two objects belonging to two



(a) (b)

Figure 5.2: (a) shows several clusters of objects. The dotted white line is the window over the cluster set with no overlap between any window. (b) Three objects shown in one cluster. The dotted white window represents the maximum motion an object may undergo.

different clusters may overlap each other since the window of objects belonging to different clusters do not overlap. The depth order permutations for each cluster is thus restricted to objects within the cluster set.

Each cluster itself may have several objects. Further reduction in depth orders is therefore desirable. For each cluster, depth order permutations that test the depth order between two objects that do not lie in each other's window are removed. Figure 5.2(b) illustrates the idea. The three objects of white (w), light gray (lg), and dark gray (dg) color, are shown with a dotted white window which is the limit of motion for each object. The total depth orders for tracking to the next frame would be $3! = 6$. However, the white object cannot overlap with the dark grey object in the next frame since there is no overlap between their motion windows. So we only need to check three depth orders $z = \{(\mathrm{w},\ \mathrm{lg}, \mathrm{dg}), (\mathrm{lg},\ \mathrm{w}, \mathrm{dg}), (\mathrm{w},\ \mathrm{dg}, \mathrm{lg})\}$. The other three depth orders $\{(\mathrm{dg},\ \mathrm{w}, \mathrm{lg}), (\mathrm{lg}, \mathrm{dg},\ \mathrm{w}), (\mathrm{dg}, \mathrm{lg},\ \mathrm{w})\}$ can be ignored since if two objects do not overlap then their position relative to each other in the depth order vector is of no consequence. Since the first three depth orders check for the relative depth order of

the white and light gray objects, and of the light gray and dark gray objets, which is sufficient information to establish the full depth ordering. The other three depth orders are redundant since the white object is assumed to not overlap with the dark gray object.

Solving the PDE (5.14) in step 3 is computationally complex for large images. This problem can be mitigated by solving the PDE in a window defined around each object cluster. Nonrigid transform is therefore computed simultaneously for only those objects that belong to the cluster set. The PDE is then solved separately for all clusters.

Chapter 6

Results

In this chapter, results are provided for different examples and scenes, with parameter values specified for the experiments.

6.1 General Description

The start of the tracking algorithm requires the initial level set segmentation be provided. In the examples presented here, the segmentation has been done manually and the level sets initialized to signed distance functions with the method provided in 5.5.2. The maximum number of iterations for the nonrigid fluid PDE is typically set to $L = 200$. The Lamé constants in the viscous fluid model $\lambda \in \mathbb{R}$ and $\mu \in \mathbb{R}^+$ control the deformation of the fluid. Increasing the value of μ corresponds to increasing the viscosity of the fluid. Roughly speaking, a larger value of μ requires a larger magnitude of force to produce the same deformation as a smaller value would in any given time step. The parameter λ when positive causes contraction of the fluid in the direction perpendicular to the direction of applied force and a negative value will cause an expansion. For all experiments, the Lamé constants are set to $\lambda = 0$ and $\mu = 1$. These are set so as to provide maximum deformability for the shape of the tracked objects and having no lateral deformation. The force scaling factor α is set such that typically the maximum velocity in the starting iterations is in the order of 10^{-1}. This leads to maximum displacement on the order of 10^0 to 10^{-1} pixels in a single iteration of the fluid PDE. The maximum motion constraint for each object is typically set to

Figure 6.1: Tracking of a taxi using the combined rigid and nonrigid model.

10 pixels or 1.5% of image size, whichever is smaller, in each direction along x and y axis. The extent of the level set reinitialization is set to the same number of pixels around the boundary of the object.

6.2 Examples

In this section, examples that demonstrate various aspects of the tracking model are presented. We start with example of a single object to test the model without occlusions. Then tracking objects through occlusion, first for single object, and later for multiple objects with both simple and complex backgrounds, are considered. All the examples, except for moving ellipses, are run with the reconstruction based measure. Later, the results of the image difference and segmentation measures are compared.

6.2.1 Taxi Sequence

Consider a relatively simple example of a taxi video[1] in Figure 6.1 used frequently in tracking related papers such as [19]. In this sequence, the tracked object is a white taxi which is making a turn on a street corner. Other moving vehicles in the video are not tracked. The results show that the tracked contour match well with the object boundary. Note that the rigid model in step 2 of Algorithm 5.3 does not have rotation parameters, but the turning motion is still captured by the nonrigid model.

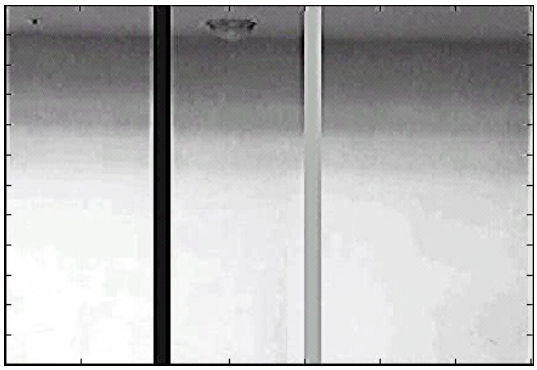
6.2.2 Man Sequence

This example shows tracking of a single person; see Figure 6.2. The test video is obtained from [48] which uses a logic segmentation model for tracking. The man enters the room from the left side and walks across it. There are also two bars present in the video. One bar matches the color of man's tracked region while the other bar is of different color. The camera is static in the actual video and captures the whole room. However, for the purpose of illustrating the contours in detail, the images in Figure 6.2(b), are zoomed and clipped to a region around the object of interest.

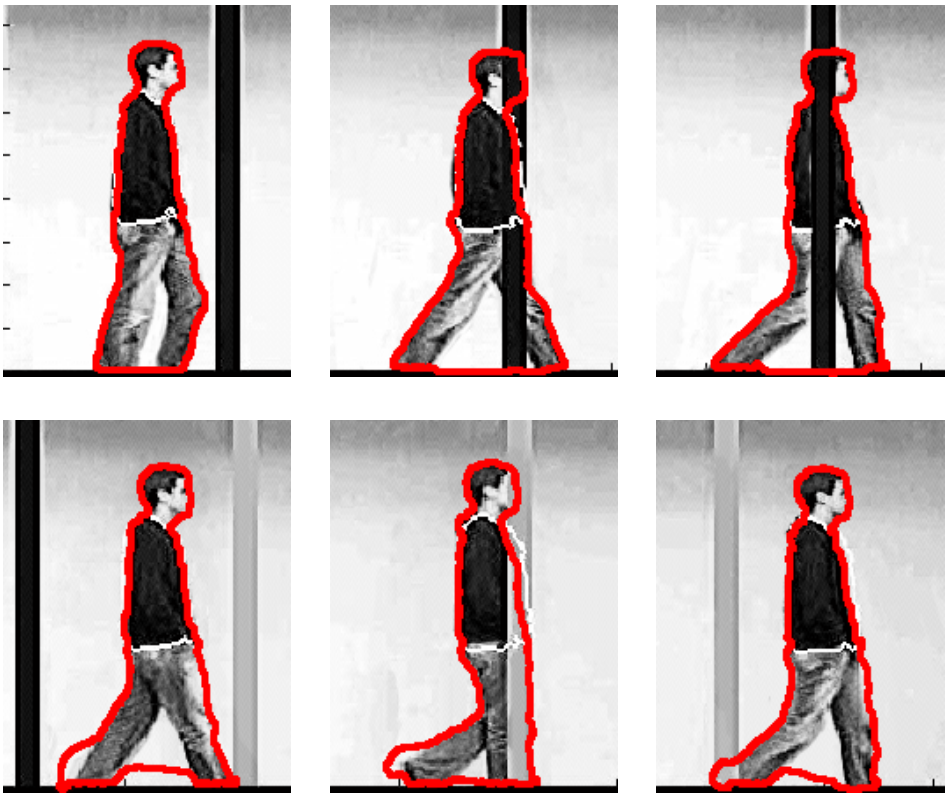
This example demonstrates the occlusion tracking capability of our model. The white boundary represents result of the logic segmentation model [48]. Whereas the white boundary only tracks the torso, we track the whole person as shown by the red boundary. This example demonstrates better results using our method, as can be seen from the seventh (top-middle), twenty sixth (bottom-middle) and forty eight (bottom-right) frame. The white boundary (their model) lies outside the man's body whereas the red contours from our model remain accurate. The contours in the occluded region correspond well to the expected shape of a person.

In the video, the legs move such that they occlude themselves when crossing each other from the camera perspective. Also, the legs have more deformation as compared to the head and torso. For these reasons part of the background in between the legs is also captured. In the twenty sixth and forty eight frame, the shape of the region between the legs is slightly different to that in earlier frames. This happens when the legs cross each other causing the region to shrink. It does not, however, affect the

[1]Image sequence courtesy - Prof. Dr. H.-H. Nagel, Image Sequence Server (http://i21www.ira.uka.de/image_sequences) (1997)



(a)



(b)

Figure 6.2: Tracking of a person walking in a room occluded by bars of two different colors. (a) shows the whole room, and images in (b) the tracked results in a window around the person. Frames shown are the first (top-left), seventh (top-middle), tenth (top-right), twenty sixth (bottom-left), thirty third (bottom-middle), and forty eight (bottom-right).

quality of the results. The tracked contour remains close to the outer boundary of the man's body, whereas the background region between the legs expands and shrinks as the legs move. In twenty sixth and thirty third frame, small errors may be observed at the back of the man's legs. During occlusion, large deformation of the legs in the occluded region is not completely captured, which causes this error. These types of errors are not corrected as the background does not change significantly from frame to frame as has been explained in Section 4.2. However, the error does not increase to affect the overall results significantly.

6.2.3 Moving Ellipses

The examples in Figure 6.3 and Figure 6.4 show several ellipses. The ellipses move and change shape as shown by the frames. The pink one moves to top left, purple moves top right, green moves right and yellow moves down. In the process, these ellipses occlude each other in complex ways. For instance in the second image, all ellipses occlude each other.

In Figure 6.3, the tracking results are obtained from Algorithm 5.3 and heuristics for z-order as discussed in Section 5.5.3 are used. In the first few frames, the pink ellipse is at a distance from the other three, so a total of $3! + 1 = 7$ depth orders are checked. In other frames where all objects are close, $4! = 24$ depth orders have to be checked. In the last few frames only one depth order is required for each of the pink and yellow ellipse, and two are required for the green and purple ellipses, i.e., a total of 4 depth orders are tested. The example shows that the depth order detection in the rigid model works correctly. Also, the contour of the object in occluded regions are close to the expected circular shapes, despite complex inter occlusions.

In Figure 6.4, the two similarity measures, reconstruction based and segmentation based, are used. Additionally, Gaussian noise has been added to the images to test the robustness of the similarity measures. The top-right images in the Figure 6.4(a) and Figure 6.4(b), shows the results are comparable for both the models. However, as the tracking progresses further as shown in bottom-left and bottom-right images, errors start to accumulate in the results from the reconstruction based measure. The segmentation based model, however, produces better tracked contours since the measure is robust to noise and error. Notice there is a slight error in the contour of the pink ellipse in the bottom-left frame of Figure 6.4(b). This and any other error introduced

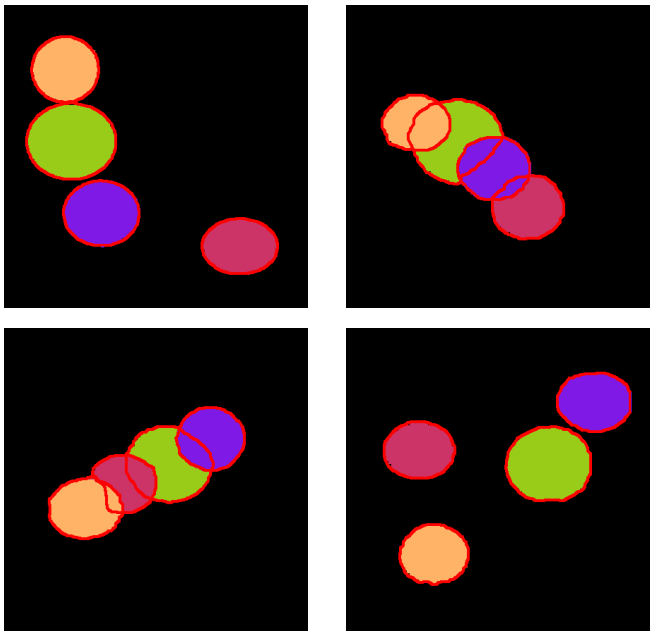
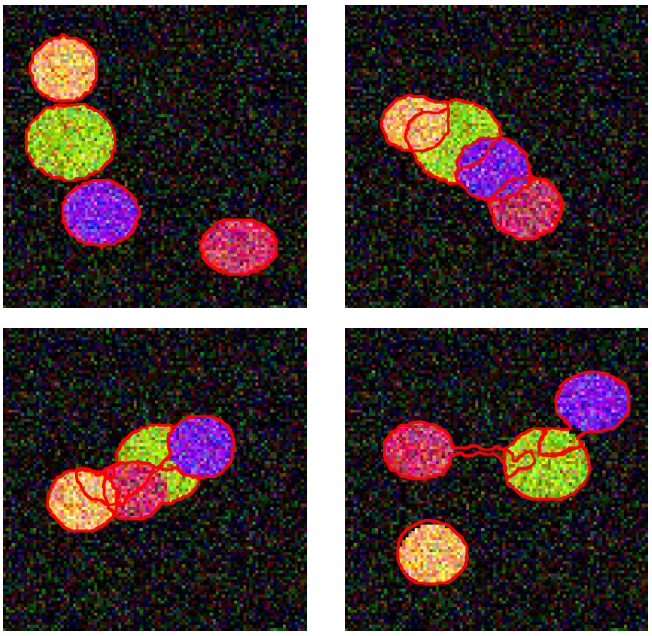


Figure 6.3: Tracking of moving and deforming ellipses using the reconstruction based measure.

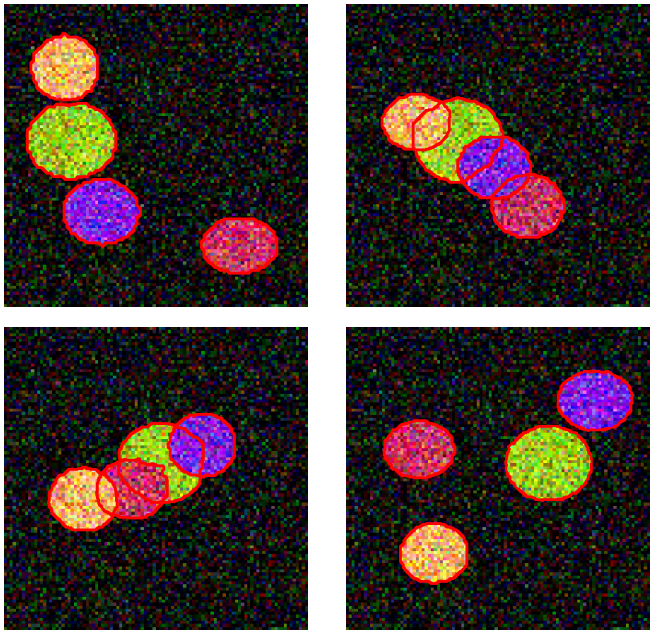
during the tracking process corrects itself as has been discussed in Section 4.2.

6.2.4 Snooker

A real wold example that is similar to the synthetic moving ellipse example, is shown in Figure 6.5. The scene here has a more complex background than the one in the synthetic example. The background is a billiard board. The white ball as it is hit by a pool cue scatters the yellow ball which in turn hits the black ball and so on. The blue ball remains stationary throughout the video. Although the balls themselves are rigid, they appear bigger as they come closer to the camera. Our tracking model successfully captures all the occlusions occuring in the video. The contours are circular in the occluded regions as expected. The first four images (top and middle rows) contain a hand and a stick which are not tracked objects. Note the contours are not circular because the shadows are also captured as part of the object. This is because the shadows are not present in the background image and they are attached to the object, so the difference between the reconstructed and target image is non zero in the



(a)



(b)

Figure 6.4: Tracking of moving and deforming ellipses with Gaussian noise, using (a) the reconstruction based measure, and (b) the segmentation based measure.

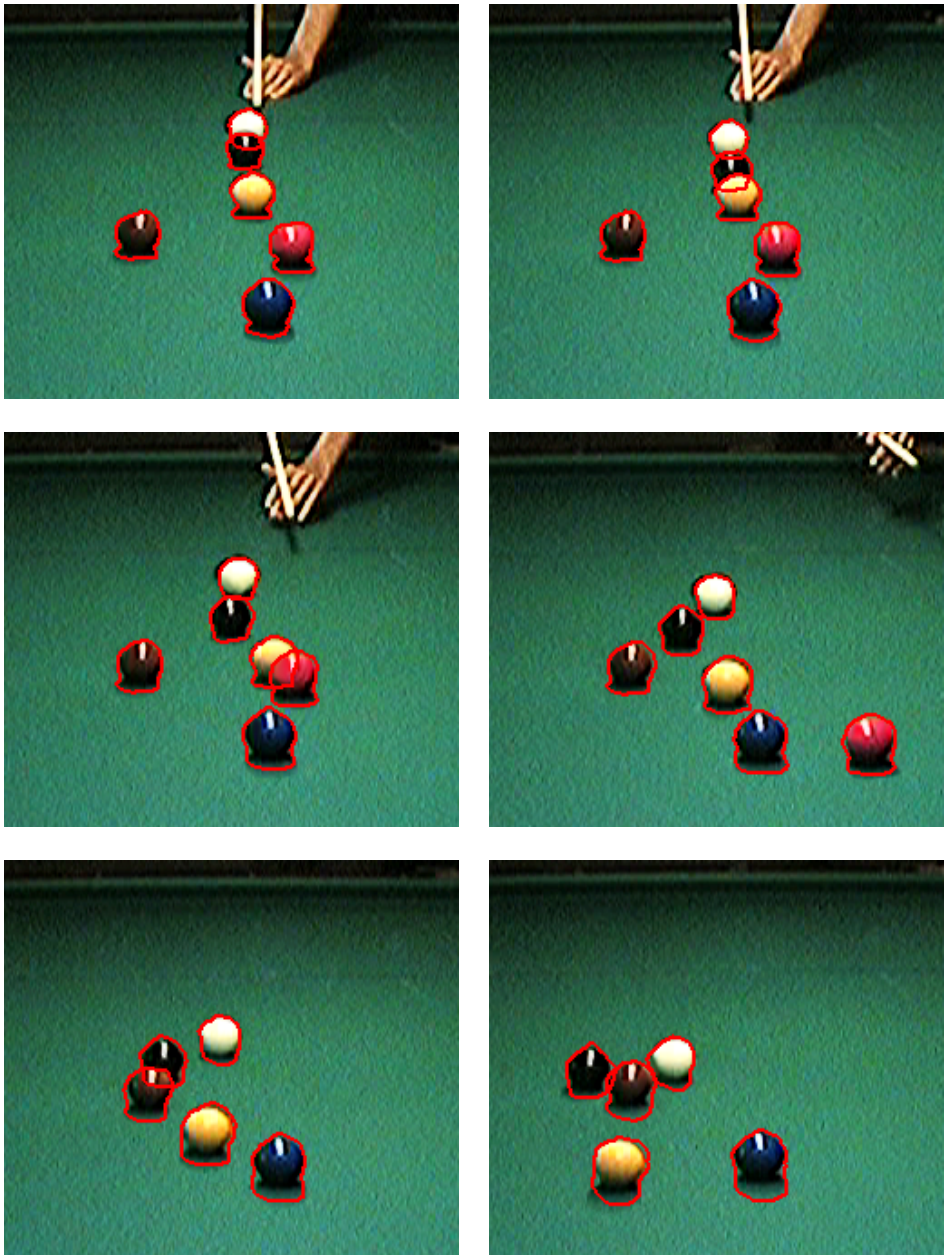


Figure 6.5: Tracking of snooker balls as they move and occlude each other. Frames are ordered in the sequence top-left, top-right, middle-left, middle-right, bottom-left and bottom-right.

region.

6.2.5 Walking people

In this example, a scene with three people walking and occluding each other is tracked; see Figure 6.6. The objects in this video have larger deformation compared to moving ellipses or the snooker example. One person in yellow shirt, whom we briefly call as person 1, is moving across the scene and occludes another person in blue jacket, whom we call person 2, who is moving away from the camera. The last person in light pink shirt, called person 3, is moving across in the direction opposite to person 1.

The tracking model captures the contour correctly of both the upper body and the legs. The contours of person 1 and 3 are quite close to the actual boundaries. The occluded region of person 2 is also captured properly and the shape of legs is retained. The dark black hair of person 1 is correctly tracked despite being in the vicinity of the dark blue jacket of person 2. Errors, however, occur in some of the frames in the contour of person 1 near the middle part of the body, which tend to increase progressively. This is because the scene has shadows which changes the luminance of the objects in certain regions. Normally these subtle changes would be captured by the model, since the variation occurs gradually from frame to frame. However, when these changes occur when the object is under occlusion, then the stored intensity templates may not remain accurate. Here, the intensity of the leg region of person 2 changes during occlusion because of moving legs and changing illumination. When the legs become unoccluded, they are captured incorrectly by person 1.

In this example, we also compare the performance of the fluid model vs the elastic model. The deformations in this video are large and nonlinear. Figure 6.7 shows the results from using the two nonrigid models. The elastic model Figure 6.7(a), is able to capture only small deformations and misses the shape changes especially near the leg regions, whereas the fluid model Figure 6.7(b), captures the shape correctly. Our choice of the fluid model is thus justified for tracking applications; see Section 4.4.2 for details.

Figure 6.6: Tracking of three moving people through occlusion. Frames 1 (top-left), 7 (top-right), 13 (bottom-left), and 18 (bottom-right) of the video are shown.

(a) (b)

Figure 6.7: Comparison of two nonrigid models, (a) elastic and (b) fluid. Frame 2 of the video is shown in both images.

6.2.6 Brightfield Cell Images

The images in Figure 6.8 are brightfield microscopic images containing four cells moving in a plate. The top-left image in the figure is the starting frame of the video. The background in these images is relatively simple. However, there are two challenges for object detection and tracking. Firstly, in many regions the contrast is poor and the interior of the cells have similar intensity values as other cells, as well as the background. The boundaries of the cell can be seen to be bright but this is not always the case as in some regions this feature is absent and in others it is not contiguous. Secondly, the cells are highly deformable and their motion is not uniform over several frames. In some frames they may move very little while in others there is a sudden increase in movement. Many cells can overlap each other and it is difficult to determine the overlapped regions and depth order even manually.

Despite these issues, the results are generally good. The global location of the cells are correctly tracked with the rigid step, which is visually most evident for cells 1 and 3. The nonrigid step produces deformation at the cell boundaries, that though not completely accurate, are still reasonable. Notice the nonrigid tracking of cell 2 is the most accurate. The bending motion of this cell has been captured correctly. In the fourth frame (top-right), cell occlusion between cells 2, 3 and 4, is also captured. Our model at the moment uses region intensity as feature, so regions of low contrast on the outer parts of cells 1 and 3 are difficult to capture. In the future, results could be improved further by using a segmentation measure desgined for brightfield cell images.

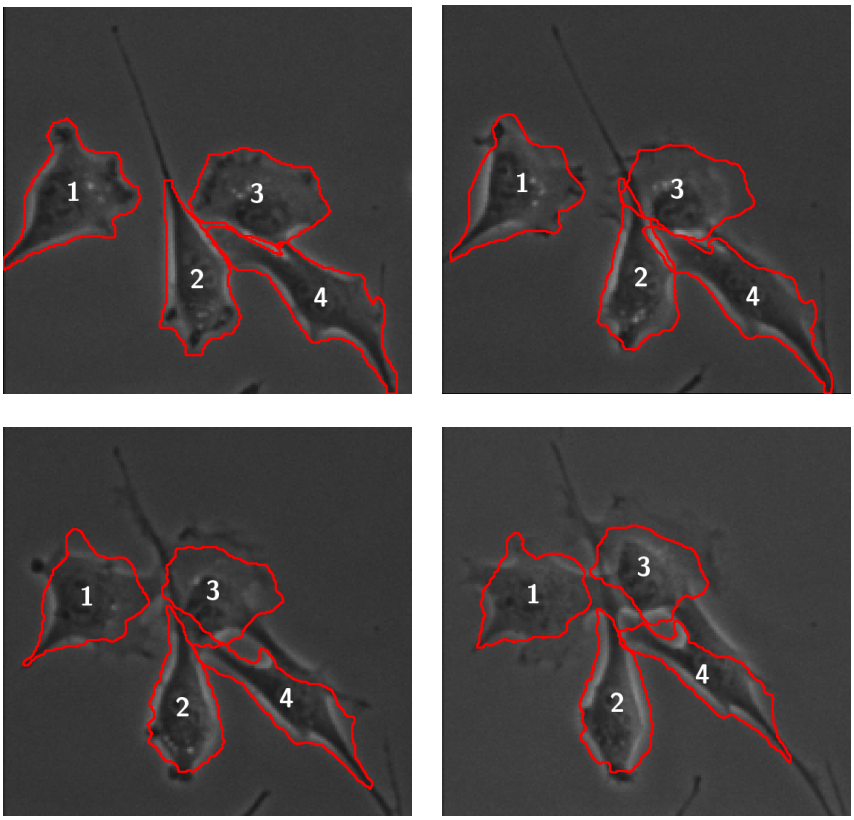


Figure 6.8: Tracking of cells in brightfield images. Frames one (top-left), four (top-right), ten (bottom-left), and fifteen (bottom-right) are displayed. The four cells have been numbered for identification.

Chapter 7

Conclusion

In this thesis, a general framework for multiple object tracking and occlusion has been presented. The main idea is of scene reconstruction using object region and intensity information, which is used to recover the object motion and depth order from frame to frame. The motion is comprised of rigid and nonrigid components. Rigid registration is used to recover the rigid motion component and nonrigid registration for the nonrigid motion. The rigid motion provides the estimate for the shape of the occluded region and the relative depth ordering of objects in the scene. The nonrigid motion captures the deformation of the object in the unoccluded regions, while simultaneously improving upon the rigid motion estimate of the object contour in the occluded region.

Our framework can also use any level set segmentation technique as a meaure for computing the object motion in place of the image difference measure. This improves the robustness of the tracker with respect to accuracy of the obtained object contours. Segmentation methods by themselves cannot capture occlusion unless some extra procedure is added to recover the occluded region, such as the one given by Yilmaz and Shah [55]. By contrast our method captures occlusion as part of the tracking model while at the same time benefitting from the improved accuracy of segmentation measures.

We have demonstrated the viability of our model with different examples. Simpler tracking examples such as the taxi and more complex ones of man and snooker, show the tracker computes motion accurately and captures the object contours, while at the same time providing a good estimate of the object shape in the occluded parts. A

synthetic example is also given to compare the increased accuracy of the segmentation based measure with the reconstruction based measure.

Our tracking model is designed for offline use so performance is not a major issue. But if desired for high resolution images, the time for solving the fluid equations can be reduced by using multigrid methods. There is scope for investigating different similarity measures like mutual information, statistical appearance models and active shape models, which can improve the robustness of our tracking model. Also, currently we do not gather the background information directly from the video. Techniques to resolve this issue can be worked upon in the future. Different deformable models can also be investigated like physically based ones such as non-linear elastic, fluid, mass transfer models, and non-physical ones, such as set symmetric difference, and spline model.

References

[1] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:641–647, 1994. 18

[2] Ravi Bansal, Lawrence H. Staib, Zhe Chen, Anand Rangarajan, Jonathan Knisely, Ravinder Nath, and James S. Duncan. Entropy-based, multiple-portal-to-3DCT registration for prostate radiotherapy using iteratively estimated segmentation. In *Proceedings of the Second International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 567–578, London, UK, 1999. Springer-Verlag. 13

[3] Marcelo Bertalmio, Guillermo Sapiro, and Gregory Randall. Morphing active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):544–549, 2000. 1, 8, 18

[4] S. Beucher and F. Meyer. The morphological approach to segmentation: The watershed transformation. In E.R. Dougherty, editor, *Mathematical Morphology in Image Processing*, pages 433–481. Marcel Dekker, 1993. 18

[5] J. Ross Beveridge, Joey Griffith, Ralf R. Kohler, Allen R. Hanson, and Edward M. Riseman. Segmenting images using localized histograms and region merging. *International Journal of Computer Vision*, 2:311–347, 1989. 17

[6] A Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin Calcutta Math Society*, 35:99–109, 1943. 6

[7] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 0:232, 1998. 1

[8] Andrew Blake and Alan Yuille, editors. *Active Vision*. MIT Press, 1992. 7

[9] M. Bro-Nielsen. *Medical Image Registration and Surgery Simulation*. PhD thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 1996. IMM-PHD-1996-25. 16, 17

[10] T J Broida and R Chellappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):90–99, 1986. 5

[11] Chaim Broit. *Optimal registration of deformed images*. PhD thesis, Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA, 1981. 16

[12] Vincent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1995. 7

[13] Tony F. Chan and Luminita A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10:266–277, 2001. 18, 19, 20, 21, 33, 44

[14] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003. 6

[15] R. Cucchiara, C. Grana, G. Tardini, and R. Vezzani. Probabilistic people tracking for occlusion handling. In *International Conference on Proceedings of the Pattern Recognition*, volume 1, pages 132–135, Washington, DC, USA, 2004. IEEE Computer Society. 6

[16] M. Isard and J. MacCormick. Bramble: a bayesian multiple-blob tracker. In *IEEE International Conference on Computer Vision*, volume 2, pages 34–41, 2001. 6

[17] Michael Isard and Andrew Blake. Condensation—conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998. 5

[18] Jeremy D. Jackson, Anthony J. Yezzi, and Stefano Soatto. Tracking deformable moving objects under severe occlusions. In *IEEE Conference on Decision and Control*, 2004. 8

[19] Jeremy D. Jackson, Anthony J. Yezzi, and Stefano Soatto. Dynamic shape and appearance modeling via moving and deforming layers. *International Journal of Computer Vision*, 79(1):71–84, 2008. 9, 70

[20] Larry Junck, John G. Moen, Gary D. Hutchins, Morton B. Brown, and David E. Kuhl. Correlation methods for the centering, rotation, and alignment of functional brain images. *Journal of Nuclear Medicine*, 31(7):1220–1226, 1990. 13

[21] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988. 7, 18

[22] C.T. Kelley. *Iterative Methods for Optimization*. SIAM, 1999. 55

[23] Sohaib Khan and Mubarak Shah. Tracking people in presence of occlusion. In *Asian Conference on Computer Vision*, pages 1132–1137, 2000. 6

[24] J. Kybic and M. Unser. Fast parametric elastic image registration. *IEEE Transactions on Image Processing*, 12:1427–1442, 2003. 16

[25] J. Lautissier, L. Legrand, A. Lalande, P. Walker, and F. Brunotte. Object tracking in medical imaging using a 2d active mesh system. In *Proceedings of the IEEE on Engineering in Medicine and Biology Society*, volume 1, pages 739–742, Sep 2003. 1

[26] F. Leymarie. Tracking deformable objects in the plane using an active contour model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):617–634, 1993. 7

[27] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 4

[28] L. Lucchese and S. K. Mitra. Unsupervised segmentation of color images based on k -means clustering in the chromaticity plane. In *Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries*, page 74, Washington, DC, USA, 1999. IEEE Computer Society. 17

[29] Frederik Maes, Andre Collignon, Dirk Vandermeulen, Guy Marchal, and Paul Suetens. Multi-modality image registration maximization of mutual information. *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, 0:0014, 1996. 13

[30] Abdol-Reza Mansouri. Region tracking via level set PDEs without motion computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):947–961, 2002. 8

[31] Jan Modersitzki. *Numerical Methods for Image Registration*. Oxford University Press, 2004. 16

[32] Wei Niu, Jiao Long, Dan Han, and Yuan-Fang Wang. Human activity detection and recognition for video surveillance. In *IEEE International Conference on Multimedia and Expo*, volume 1, pages 719–722, Jun 2004. 1

[33] Nikos Paragios and Rachid Deriche. Geodesic active regions for supervised texture segmentation. *IEEE International Conference on Computer Vision*, 2:926, 1999. 18

[34] Nikos Paragios and Rachid Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000. 7, 18

[35] Nikos Paragios and Rachid Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, 46(3):223–247, 2002. 18

[36] Christopher Rasmussen and Gregory D. Hager. Probabilistic data association methods for tracking complex visual objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):560–576, 2001. 5

[37] Yogesh Rathi, Namrata Vaswani, Allen Tannenbaum, and Anthony Yezzi. Tracking deforming objects using particle filtering for geometric active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(28):1470–1475, 2007. 8

[38] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, Dec 1979. 5

[39] K. Rohr. *Landmark-Based Image Analysis: Using Geometric and Intensity Models*. Kluwer Academic Publishers, 2001. 13

[40] Remi Ronfard. Region-based strategies for active contour models. *International Journal of Computer Vision*, 13(2):229–251, 1994. 7

[41] I. K. Sethi and Ramesh Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):56–73, 1987. 1

[42] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science (Cambridge on Applied and Computational Mathematics)*. Cambridge University Press, 2nd edition, June 1999. 18, 20, 65

[43] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 731–737, Jun 1997. 18

[44] Jianbo Shi and Carlo Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593 – 600, 1994. 4, 6

[45] Hai Tao, H.S. Sawhney, and R. Kumar. Object tracking with bayesian estimation of dynamic layer representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):75–89, Jan 2002. 9

[46] Shutong Tse, Laura Bradbury, Justin W.L. Wan, Haig Djambazian, Robert Sladek, and Thomas Hudson. A combined watershed and level set method for segmentation of brightfield cell images. In Josien P. W. Pluim and Benoit M. Dawant, editors, *Medical Imaging 2009: Image Processing*, volume 7259, page 72593G. SPIE, 2009. 1

[47] C.J. Veenman, M.J.T. Reinders, and E. Backer. Resolving motion correspondence for densely moving points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):54–72, Jan 2001. 5

[48] J.H. von Brecht, S.R. Thiruvenkadam, and T.F. Chan. Occlusion tracking using logic models. In *Signal and Image Processing*. Acta Press, 2007. 70

[49] J.Y.A. Wang and E.H. Adelson. Layered representation for motion analysis. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 361–366, Jun 1993. 9

[50] Robert Weinstock. *Calculus of Variations - With Applications To Physics And Engineering*. Weinstock Press, 2008. 41

[51] Yu-Te Wu, Takeo Kanade, C.-C. Li, and Jeffrey Cohn. Image registration using wavelet-based motion model. *International Journal of Computer Vision*, 38:129 – 152, 2000. 16

[52] Alper Yilmaz Xin, Xin Li, and Mubarak Shah. Object contour tracking using level sets. In *Asian Conference on Computer Vision*, 2004. 8

[53] Anthony J. Yezzi and Stefano Soatto. Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images. *International Journal of Computer Vision*, 53:153–167, 2003. 8

[54] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):13, 2006. 4

[55] Alper Yilmaz, Xin Li, and Mubarak Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, 2004. 8, 18, 36, 37, 80

[56] Alistair A. Young. Model tags: direct three-dimensional tracking of heart wall motion from tagged magnetic resonance images. *Medical Image Analysis*, 3(4):361 – 372, 1999. 1

[57] Song Chun Zhu and Alan Yuille. Region competition : Unifying snakes, region growing and bayes/mdl for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996. 7, 18

[58] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977 – 1000, 2003. 13