A METHOD FOR ESTIMATING MOTIONS OF CONTOURS
WITH AN APPLICATION TO GAIT RECOGNITION

by

Sam Gelman

A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
Computer Science

Committee:

_____	Dr. Zoran Durić, Thesis Director
_____	Dr. Wallace Lawson, Committee Member
_____	Dr. Lynn Gerber, Committee Member
_____	Dr. Sanjeev Setia, Chairman, Department of Computer Science
_____	Dr. Kenneth Ball, Dean Volgenau School of Engineering
Date: _____	Summer Semester 2016 George Mason University Fairfax, VA

A Method for Estimating Motions of Contours with an Application to Gait Recognition

A thesis submitted in partial fulfillment of the requirements for the degree of
Master of Science at George Mason University

By

Sam Gelman
Bachelor of Science
George Mason University, 2014

Director: Dr. Zoran Durić, Associate Professor
Department of Computer Science

Summer Semester 2016
George Mason University
Fairfax, VA

Dedication

I dedicate this thesis to my parents for their support and encouragement, which has enabled me to pursue my dreams to my fullest potential.

Acknowledgments

I want to express my sincere gratitude to Dr. Zoran Duric, my advisor and mentor over the last six years. He introduced me to the world of research and guided me on the path to this work. I would also like to thank my committee members, Dr. Lynn Gerber and Dr. Wallace Lawson. Dr. Gerber has advised me since my first project, and Dr. Lawson has greatly encouraged me to pursue this work over the last year. A special thanks goes to my parents, grandparents, and my family for their endless support and encouragement. Thank you Ben, Sasha, Grant, Roman, and Stephanie. I would also like to thank my closest friends Hannah King, Michael Bowen and Mackenzie Sweeney. I greatly appreciate your support in my endeavors. Finally, I extend my gratitude to my friends at George Mason's Autonomous Robotics Laboratory: David Bagheri, Kevin Andrea, Raven Russell, David Fleming, Sam McKay, Chris Vo, Keith Sullivan. Thank you for the friendship, advice, and laughter over the years.

Table of Contents

List of Tables

List of Figures

Abstract

A METHOD FOR ESTIMATING MOTIONS OF CONTOURS WITH AN APPLICATION TO GAIT RECOGNITION

Sam Gelman

George Mason University, 2016

Thesis Director: Dr. Zoran Durić

In this thesis, I propose a novel method for estimating motions between image contours. The method is fast and can handle both small and large displacements. It uses the distance transform and its gradients to estimate correspondences between points. The distance transform of a binary contour represents the distance of each pixel to the nearest contour pixel. The gradient of the distance transform points in the direction normal to the contour. By combining unit vectors obtained from the gradient with the original distance transform, the method produces vectors that correspond to the normal displacement between pairs of contours. This method can then be extended to compute true motion near corners as well as parameterized motion models.

Experiments on various shape contours show the method's efficacy in computing normal displacements. Cases that do not correspond to normal motion are analyzed and corrected through the use of contour normals and motion models. The method is also applied to gait recognition, the goal of which is to identify people in videos based on their unique walking pattern. Many gait recognition methods operate on sequences of binary silhouettes. Since contours are easily obtainable from silhouettes, the proposed method is well-suited to the task. I describe two representations, the Histogram of Motion and the Edge Motion Vector,

that allow for the comparison of contour motion between frames and sequences. These representations are tested on a large gait recognition database and achieve rank 1 and rank 5 performance that is comparable to the state of the art. The success of the method on gait recognition shows it is useful as a standalone representation, but it can also be used to improve other techniques and for other applications - this is left to future work.

Chapter 1: Introduction

The goal of motion estimation is to compute displacement between corresponding features in pairs of images. Motion estimation has a long history of research dating back to the seminal works of Horn and Schunck [1] and Lucas and Kanade [2]. It has been used in a wide range of application domains including video editing, action recognition, and person identification. The motions of interest in most applications are the true motions of objects, e.g. people, cars, robots. However, actual motion in three-dimensional space does not necessarily correspond to apparent motion in two-dimensional images. Apparent motion can be caused by a variety of factors including the changes in the lighting of a scene. Motion estimation techniques are therefore concerned with finding the apparent projection in the two-dimensional image of the actual motion in a three-dimensional scene.

Motion estimation techniques can be roughly divided into two categories: dense and sparse. Dense techniques are concerned with finding a motion vector for each pixel in an image. The resulting motion field is referred to as optical flow. These techniques establish correspondences between pixel regions by assuming that the intensity of pixel regions remains constant during displacement. This assumption is only valid for small translational displacements. Conversely, sparse techniques are concerned with finding motion vectors between a small number of select features. Once features are matched, displacement can be computed for arbitrarily large distances. Some recent work on motion estimation has focused on combining dense and sparse techniques.

Motion can also be represented parametrically based on motion vectors computed from dense or sparse techniques. Such representations are useful for handling errors and compactly describing motion. Parametric motion models are subject to a trade-off between complexity and representativeness. A good balance between these factors usually limits

the use of motion models to rigid objects. One of the most commonly used motion models for describing two-dimensional motion is the affine motion model. The affine model is a six-parameter model that represents translation, rotation, scale, and shear. It assumes the motion is that of a three-dimensional rigid planar surface projected orthographically onto the image plane. In many applications, sparse and dense motion fields, along with parametric motion models, are a basis from which meaningful information is extracted.

In this thesis, I propose a new method for estimating motions of object contours. The method uses the distance transform and its gradients to establish correspondences between point sets. The distance transform of a contour image computes, for each image pixel, the distance to the nearest contour pixel. The gradient of the distance transform points in the direction normal to the contour. By combining unit vectors obtained from those gradients with the original values from the distance transform, the method produces vectors that correspond to the normal displacement between pairs of contours.

The proposed method has several advantages compared to traditional optical flow techniques. It computes motion that is denser than motion computed by sparse techniques, but unlike dense techniques, it only computes motion for pixels of interest. The proposed approach can handle both small and large displacements, and it is faster than traditional optical flow techniques. Furthermore, the method can be used as a standalone representation or to estimate parameterized motion models.

In addition to testing the method on images of shape contours and using it to compute motion models, I show a practical utilization of the method by applying it to gait recognition. The goal of gait recognition is to identify people in videos based on their unique walking pattern. Many gait recognition methods are designed to work on sequences of binary silhouette images. Since contours are easily extracted from silhouette images, the proposed method is well-suited to the task. I show that my approach to gait recognition achieves rank 1 and rank 5 performance that is comparable to the current state-of-the-art when applied to a large gait recognition database.

The remainder of this thesis is organized as follows. The second chapter describes related

work on motion estimation, distance transforms, and gait recognition. The third chapter describes the method for estimating motions of contours in detail. The fourth chapter shows how the method can be applied to gait recognition. The fifth chapter presents a conclusion and discusses future work.

Chapter 2: Related Work

Previous work on motion estimation has goals similar to the method presented in this thesis. However, previous work on distance transforms is most closely related when it comes to methodology. In the following sections, I review related work for both of these topics, as well as related work on gait recognition.

2.1 Motion Estimation

Traditional dense flow techniques use an energy minimization framework first introduced by Horn and Schunck [1]. These approaches establish correspondences between points by assuming constancy of some image property and enforcing a smoothness constraint on the motion. Lucas and Kanade [2] introduced a coarse to fine warping scheme to deal with large displacements that is also a major component of most flow estimation techniques. Many improvements have been made to the original formulations, including the use of robust statistics to handle outliers [3] and a gradient constancy assumption [4] in addition to the original brightness constancy assumed by Horn and Schunck. Further improvements were made in the derivative computation and through the use of different penalty terms to enforce smoothness [5]. These techniques have demonstrated increasing accuracy on optical flow benchmarks, but they are usually quite slow and cannot handle large displacements. Sparse descriptor matching approaches such as KLT [6] are faster and better at handling large displacements, but establish few correspondences and are more susceptible to outliers. Brox et al. [7] integrated point correspondences from descriptor matching into an energy minimization framework to compute large displacement dense optical flow.

Methods for computing motion between contours differ methodologically from methods described above, but they are faced with similar challenges. Motion along straight lines

cannot be determined without integrating information from features such as corners or textures. An early method for contour-based motion estimation [8] computed motion at corners and propagated that motion along contours. Later methods [9–11] matched contours using a criterion of minimum curvature differences. Epipolar geometric constraints obtained from corner matching combined with contour end point constraints and contour distance measures have also been used to match contours [12]. Liu et al. [13] computed local estimates at edgelets and boundary fragments. Global motion estimates were then derived by grouping edgelets and fragments into contours using a graphical model. In [14], contours were divided into left-pointing and right-pointing parts and the distance transform was used to estimate displacement. My method is an improvement of this work.

2.2 Distance Transforms

Distance transforms have been used extensively in computer vision, image processing, and pattern recognition. Several distance transforms and their applications were discussed in [15]. Efficient methods for computing Euclidean distance have appeared more recently [16]. The method proposed in this thesis was implemented using the distance transform algorithm described in [17].

One common application of distance transforms is for image matching. Hierarchical Chamfer Matching [18] convolves a binary contour of an object with a distance transform image computed from a template. Test contours must be aligned for matching to work well. Since matching is computationally expensive at high resolutions, a series of matches starting at low resolutions was employed. In [19] Hausdorf distance was used to match images. The Hausdorf distance was efficiently computed using the distance transform. Distance transform derivatives have also been used to design efficient algorithms for robust matching of points sets [20].

2.3 Gait Recognition

Related work on gait recognition can be divided into two categories: model-based and model-free. Model-based approaches incorporate an underlying model for gait recognition. For example, these approaches may track individual limb segments, create a skeleton model of the subject, or measure the stride length and cadence of gait. They rely on videos as well as other methods, such as 3D marker-based motion capture systems, to collect data [21]. Conversely, model-free approaches do not use an underlying model and are typically based on analysis of silhouette images. These approaches have lower computational costs than model-based methods, but are usually less robust to viewpoint and scale [21]. In this thesis, I propose a model-free approach based on the proposed method for estimating motion between contours.

Model-free approaches use representations that exploit both shape and motion features. There are a number of approaches inspired by Motion Energy Images and Motion History Images [22]. The Gait Energy Image (GEI) [23] is a grey-level average of select binary silhouettes from a sequence. This representation is robust to noise, but it ignores sequential information between frames. Several improvements to the GEI have been proposed. The Gait History Image (GHI) [24] was used to retain sequential information. The Frame Difference Energy Image (FDEI) [25] was proposed to suppress influence of silhouette incompleteness. The Gait Entropy Image (GEnI) [26] was proposed to be robust to unknown covariate conditions that change appearance of silhouettes. The Active Energy Image (AEI) [27] sought to address the problems of low quality silhouettes and insufficient dynamic characteristics by focusing explicitly on active regions. The Chrono-Gait Image (CGI) [28] preserves temporal information by encoding gait information into a multi-channel image. These approaches use motion and shape implicitly as part of the representation.

There are several approaches that compute motion vectors explicitly. Bashir et al. [29] compute optical flow on intensity images and generate five motion descriptors using the

resulting flow field. Flow values are discretized into binary representations, thus magnitude and direction of motion are not compared directly. The Gait Flow Image (GFI) [30] computes Horn and Schunck optical flow on silhouette images. This approach uses the magnitude of motion as a threshold to generate binary images that are averaged together similar to the Gait Energy Image. Direction of motion is discarded. The Flow Histogram Energy Image (FHEI) [31] computes Lucas and Kanade optical flow on binary silhouettes. Histograms of flow are used to represent motion per frame, then a sequence-level representation is computed by averaging frames together.

Chapter 3: Estimating Displacements Using the Distance Transform

The proposed method estimates displacements between image contours. Contours can be obtained from a variety of sources including silhouette boundaries, image edges, and boundaries between layers in depth images. For sample images in this chapter, I use the contours of synthetically generated shapes. These shapes are pictured in Figure 3.1. The term "original contour" to refers to the initial contour and the term "displaced contour" refers to the contour after it has moved. The method works under the assumption that there is sufficient similarity between the original and displaced contours so that computing motion between them is meaningful.

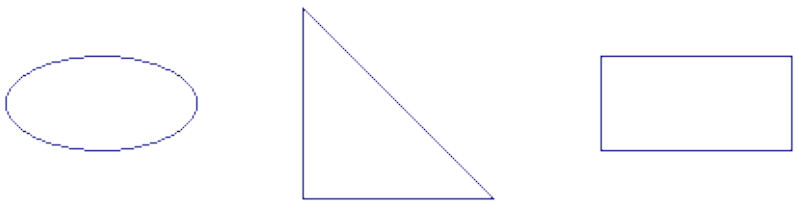


Figure 3.1: Basic shape contours used throughout this chapter. Left: ellipse. Center: triangle. Right: rectangle.

The basic method uses the distance transform and its gradients to establish correspondences between the original and displaced contours. Although this approach is very fast (it is linear in the number of pixels around the bounding box of the contour), the estimated displacements are not consistent with true motion. Rather, the displacements correspond to motion in a direction normal to the displaced contour, i.e. normal flow. This result is a

consequence of the methodology used to compute displacements. As I will show, the basic method can be improved to compute true motion near corners and points of high curvature. However, like all motion estimation techniques, the proposed method is still susceptible to the aperture problem. The aperture problem refers to the fact that the true global motion of an object cannot be determined when only looking at local motions. Figure 3.2 is a classic depiction of the aperture problem. I will show how the displacements computed using the proposed method can be integrated over an entire object in the form of a motion model, which yields true motion.

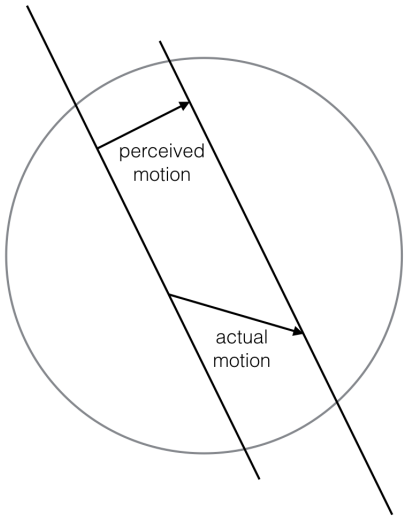


Figure 3.2: The aperture problem. Lateral motions along stretches of straight or low-curvature contours cannot be determined without integrating motion from a larger area that contains corners or other features. In these cases, only the normal component of motion can be computed.

For sample images in this chapter, I use synthetic motions of the shapes presented in Figure 3.1. Synthetic data are useful for analysis of the method as it provides a ground truth against which it is possible to compare estimated displacements. Synthetic motions are generated by applying affine transformations with known parameters to the original shape contours. The parameters allow for the control of the rotation, scale, and translation of the displacements. For analysis of the method's output, and to understand intricacies related

to the distance transform, it is also useful to define the concepts of "inside" and "outside" of a contour. According to the Jordan curve theorem [32], every simple closed curve divides the plane into an interior region bounded by the curve and an exterior region containing all other points. Any continuous path from one region to the other region intersects with the contour.

The remainder of the chapter is organized as follows. In Section 3.1, I review the distance transform and its gradients, which are the foundations of the method. In Section 3.2, I explain the method and thoroughly analyze its output. In Section 3.3, I present an improvement to the method which corrects certain errors when dealing with large displacements. In Section 3.4, I show how the method can be used as a basis to compute affine motion models.

3.1 Distance Transform and Its Gradients

3.1.1 Distance Transform

I use the definition of distance transform given in [17]. The distance transform of a binary image specifies, for each pixel, the distance to the nearest non-zero pixel. Let $\mathcal{G}$ be a regular grid and $P \subseteq \mathcal{G}$ a set of points on the grid. The distance transform associates to each grid location the distance to the nearest point in P,

$$\mathcal{D}_P(p) = \min_{q \in \mathcal{G}} (d(p,q) + \mathbf{1}(q)),$$

where $\mathbf{1}(q)$ is an indicator function for membership in P,

$$\mathbf{1}(q) = \begin{cases} 0 & \text{if } q \in P \\ \infty & \text{otherwise} \end{cases}$$

and $d(p,q)$ is the Euclidean distance, i.e. $d(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$.

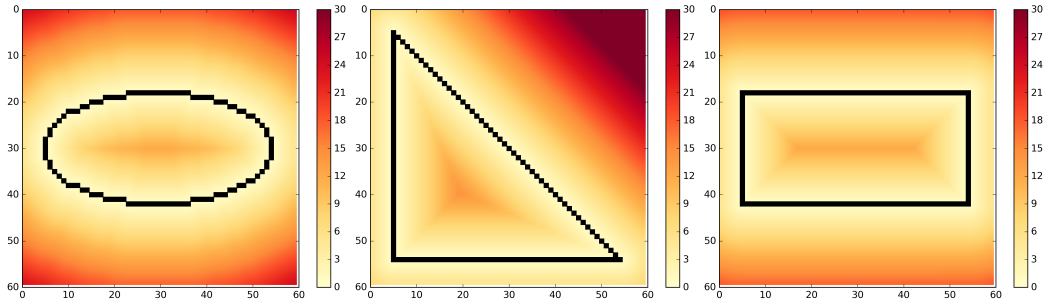


Figure 3.3: Examples of the distance transform computed for images of shape contours. Distance from the contour is represented using shading, where darker shades correspond to further distances.

Figure 3.3 shows the distance transform computed for images of shape contours. On the outside of the contours, the distance transform increases smoothly in all directions. On the inside of the contours, the distance transform increases until it reaches a ridge, i.e. a point equidistant from the contours on the opposing sides. The ridge represents a local maximum in the distance transform, and for closed contours, the ridge corresponds to the medial axis of the shape. As I show in the following sections, the ridge formed by the distance transform is an important factor when estimating displacements.

3.1.2 Gradient of the Distance Transform

The distance transform $\mathcal{D}_P(\cdot)$ is a smooth function on $\mathcal{G}$ and therefore has a gradient at each point $q \in \mathcal{G}$. The gradient

$$\nabla \mathcal{D}_P(q) = \frac{\partial \mathcal{D}_P(q)}{\partial x}\vec{i} + \frac{\partial \mathcal{D}_P(q)}{\partial y}\vec{j} \tag{3.1}$$

can be computed at all points in $\mathcal{G}$, but it is most meaningful at points for which $\mathcal{D}_P(q) \neq 0$.

Figure 3.4 shows the gradient of the distance transform represented as a vector field. On the outside of the contour, the gradient vectors are normal to and point away from the contour. The same is true on the inside of the contour, except for on and around the ridge formed by the distance transform. Gradient vectors on and around the ridge are either

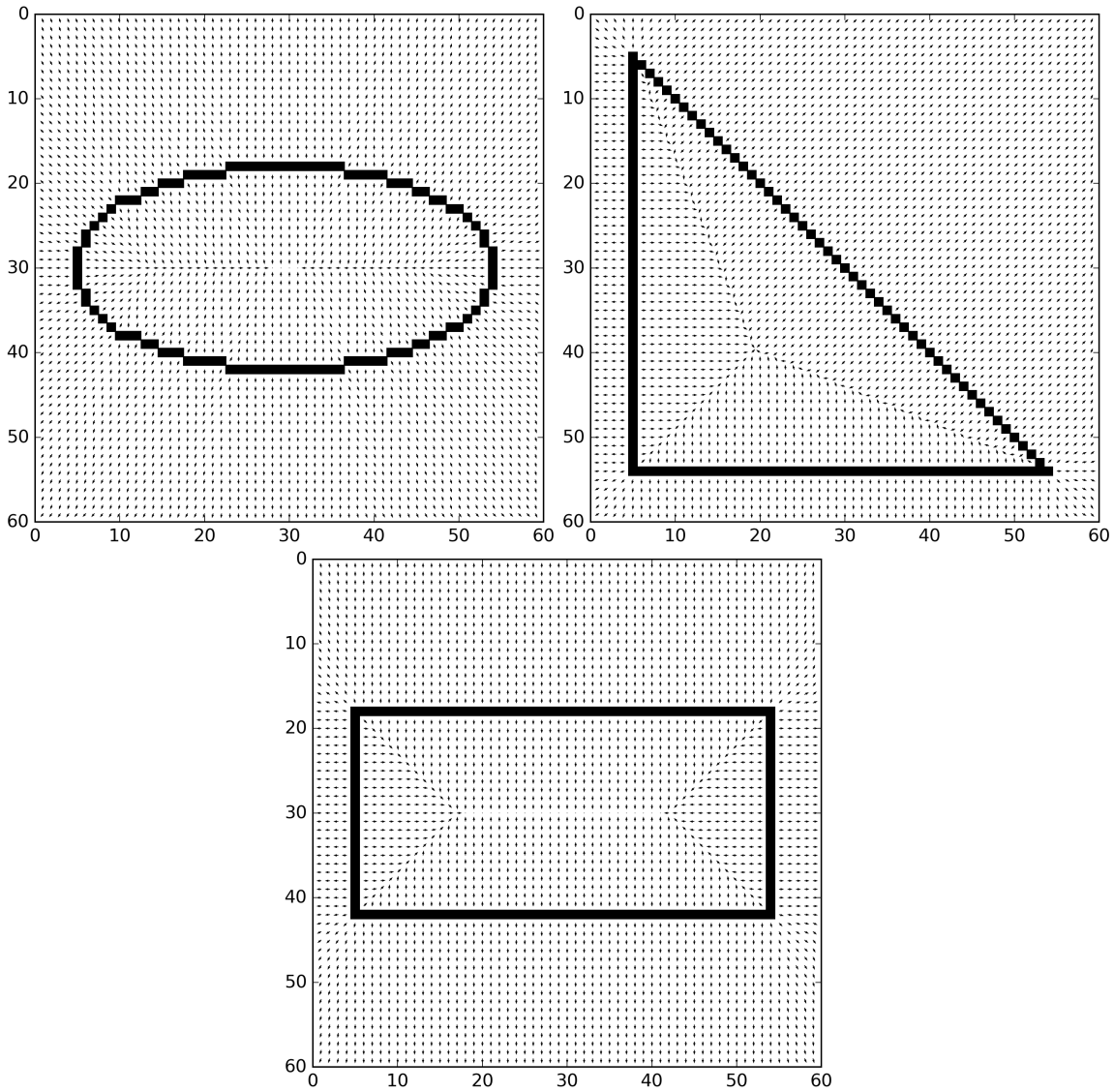


Figure 3.4: The gradient of the distance transform represented as a vector field.

zero, parallel to the medial axis, or not easily categorized. The ridge also represents the partition at which gradient vectors on the inside of the contour change from being normal to one side of the contour to being normal to a different side of the contour.

3.2 Estimating Displacements Using the Distance Transform

The observations from the previous section can be used to compute a vector pointing from any point $q \in \mathcal{G}$ to the nearest point $p \in P$. If $\|\nabla \mathcal{D}_P(q)\| > 0$ then vector

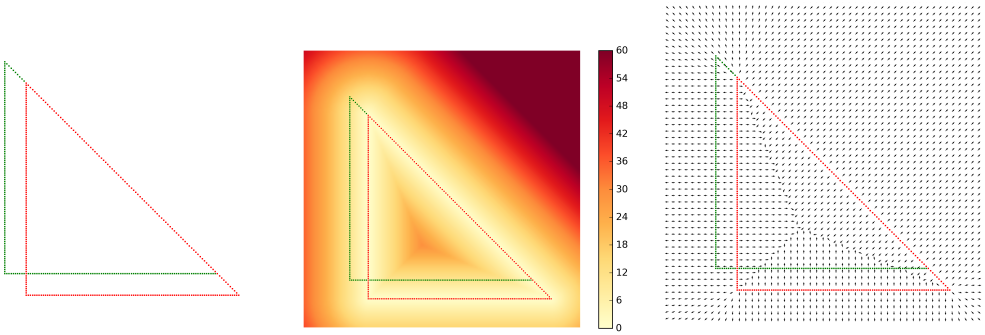
$$\vec{v}_P(q) = -\mathcal{D}_P(q) \frac{\nabla \mathcal{D}_P(q)}{\|\nabla \mathcal{D}_P(q)\|} \tag{3.2}$$

corresponds to an approximate displacement between q and some point $p \in P$. For any binary set $Q \subset \mathcal{G}$ we can compute a displacement field for all points $q \in Q$ using Eq. 3.2.

Figure 3.5 shows an overview of the method for two contours c_1 (original) and c_2 (displaced). First, the gradient of the distance transform is computed with respect to c_2. The gradient is then negated so that the gradient vectors point toward c_2 rather than away from it. Finally, the vectors at the location of c_1 are scaled by c_1's distance to c_2.

Figure 3.6 illustrates displacements computed using the proposed method. Most displacements correspond to normal flow, i.e. motion in the direction normal to the displaced contour. Some displacements correspond to true motion. The remaining displacements fall into two categories. They either point to the wrong part of the contour, as in the bottom right corner of the triangle in Figure 3.6, or they point to the correct contour but not in the direction of normal flow. These errors stem from the ridge formed by the distance transform.

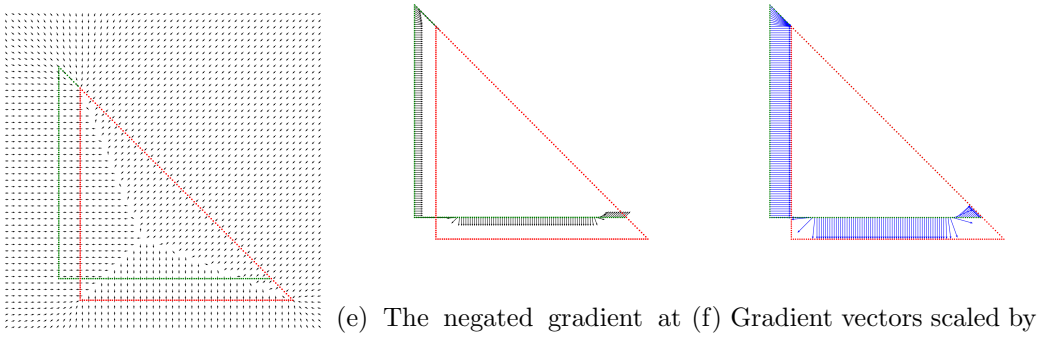
As noted above, the distance transform is calculated with respect to the displaced contour. Motion vectors can then originate from either inside or outside the displaced contour. Figure 3.7 highlights which vectors originate inside the contour and which ones originate outside the contour.



(a) The two contours for which we wish to calculate displacement

(b) The distance transform with respect to the displaced contour

(c) The gradient of the distance transform with respect to the displaced contour



(d) The negated gradient

(e) The negated gradient at the points corresponding to the original contour

(f) Gradient vectors scaled by each point's distance to the displaced contour

Figure 3.5: Overview of the proposed method.

3.2.1 Analysis of Vectors Originating Inside the Contour

Vectors originating inside the contour are subject to the effects of the ridges and the medial axes of the shape. When a point from the original contour falls on a ridge, the resulting displacement vector points in the direction of the ridge and does not correspond to true motion or normal flow. When a point from the original contour crosses the medial axis of the shape, it signifies that the point is closer to an opposing side of the contour rather than the corresponding one. The resulting vector points to the wrong part of the displaced contour.

The proportion of vectors affected by the distance transform ridge depends on the shape

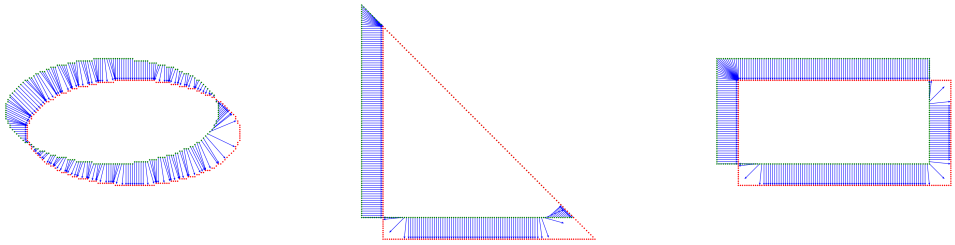


Figure 3.6: Displacements computed using the proposed method represented as vectors. The original contours are green, and the displaced contours are red.

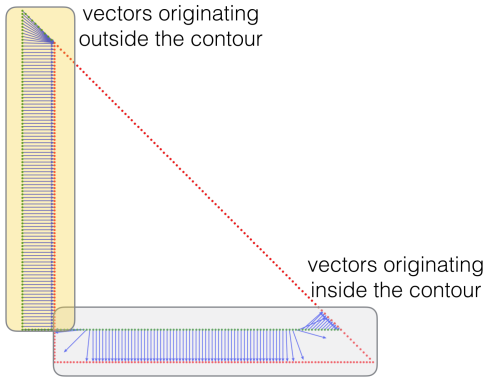


Figure 3.7: Vectors originating inside vs. outside of the displaced contour.

(specifically, the curvature of the shape) and the magnitude of displacement. At corners, points of high curvature, and for slim shapes, the ridge is very close to the opposing contours. This means that even small displacements could result in the original contour crossing the ridge, resulting in mismatched displacement vectors. This is generally not a problem along stretches of low curvature points.

3.2.2 Analysis of Vectors Originating Outside the Contour

Since the distance transform increases smoothly on the outside of the contour, vectors originating outside of the contour mostly correspond to normal flow. There are still instances where vectors point to the wrong part of the contour. An example of such an instance is

the bottom left of the triangle in Figure 3.6. The vectors from the bottom contour of the original triangle point to the left contour of the displaced triangle. Although the vectors correspond to normal flow, they do not represent true motion.

Finally, as evidenced by the ellipse in Figure 3.6, vectors sometimes bunch up around certain points rather than distributing evenly along the displaced contour. This phenomenon is due to points of the contour jutting out from the surrounding contour, and it is a common occurrence when the method is applied to real data. A simple gaussian blur can smooth the vectors so that they distribute evenly.

3.3 Using Contour Normals to Correct Displacements

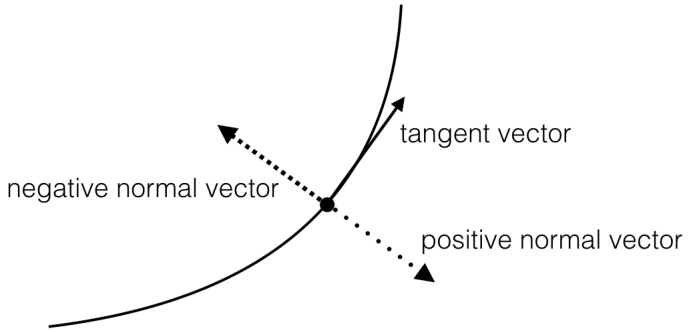


Figure 3.8: Contour normal vectors. The normal vectors are orthogonal to the tangent vector at any given point.

Some of the mismatches described in the previous section can be corrected by factoring in contour normals. For any given point along a contour, there is a tangent vector and two vectors normal to the tangent vector. The normal vectors point in opposite directions. For closed contours, the normal vectors pointing outside the contour can be defined as positive, and the normal vectors pointing inside the contour can be defined as negative. For open contours, either normal vector can be defined as positive or negative so long as the labeling is consistent. Figure 3.8 depicts positive and negative contour normals. When the term contour normal is used in this section, it can refer to either the positive or negative normal,

so long as the labeling is consistent between contours. In the context of silhouette images, a contour normal can be defined as the gradient vector at a point along a contour.

When computing displacements between contours, it is clear that corresponding points should have similar contour normals. However, this aspect is not captured inherently using the distance transform. Vectors computed using the proposed method usually point toward the nearest contour, even when the best matching contour in terms of contour normals might be further away. Figure 3.9 shows how contour normals might differ at the origin and destination of vectors computed using the proposed method. In this section, I describe a modification to the proposed method that factors in contour normals. The modified method often results in vectors that correspond to true motion, especially near corners and points of high curvature.

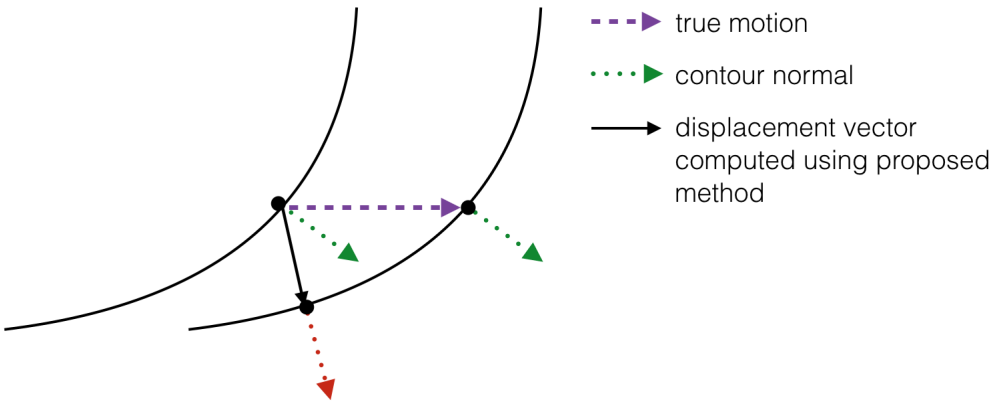


Figure 3.9: A displaced contour with contour normals depicted at the origin, the destination of the estimated displacement vector, and the destination of the true displacement vector.

Assume we have an original contour Q and a displaced contour P. The modified method begins by computing displacement vectors between Q and P as described in the previous section, without factoring in contour normals. Cases of contour normal mismatch are detected by comparing contour normals at the origin $q \in Q$ and destination $p \in P$ of each displacement vector. Let the contour normals at those points be $\vec{n}_p$ and $\vec{n}_q$. If for some

vectors the difference in the directions of the normals at the origin and destination is greater than a threshold, $|\angle(\vec{n}_p, \vec{n}_q)| < \alpha_0$, those vectors are marked as incorrect and added to subset Q^1. For all points $q \in Q^1$, we find all points $P^1(q) \subset P$ which have similar orientations and are within some distance d_0 from q. The original method is then re-applied to each point $q \in Q^1$ and the corresponding set $P^1(q) \subset P$.

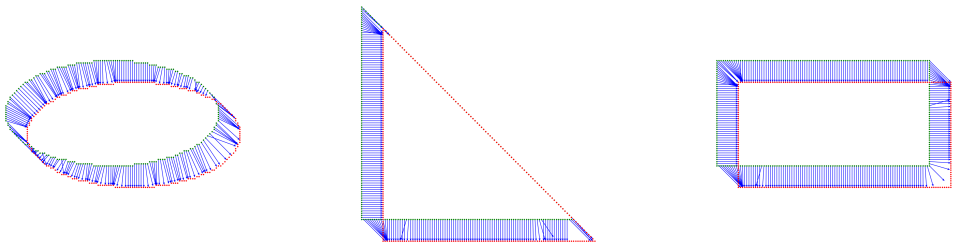


Figure 3.10: Displacements computed using the proposed method factoring in contour normals.

Figure 3.10 shows displacements computed using the modified method. Some of the corrected displacements now correspond to true motion, while others correspond to normal flow. There are some displacement vectors that are still incorrect due to being on a ridge and not having a difference in contour normals.

Although factoring in contour normals improves the results, the method is still susceptible to the aperture problem. In relation to the proposed method, the aperture problem means the lateral motions along stretches of low-curvature contours cannot be detected without integrating motion vectors over a larger area. To find true motion, we need the information obtained from features like corners and high curvature points. When dealing with rigid objects, one approach to integrating motion vectors is to compute affine motion models over the entire object.

3.4 Motion Models

As discussed in previous sections, the proposed method results in vectors that roughly correspond to normal flow. To find true motion, the vectors can be integrated over the entire shape in the form of a motion model. Motion models can have differing numbers of parameters. The more parameters used, the more expressive power the model has. A two parameter model can only represent translation. A six parameter model, typically referred to as an affine model, can represent translation, rotation, scale, and shear. One downside of using more parameters is that more vectors are required to accurately estimate the parameters. For example, say the true motion of an object is a simple translation. Both two and six parameter models can represent the motion, but in practice, factoring in possible noise in computed vectors, the two parameter estimation tends to be more accurate.

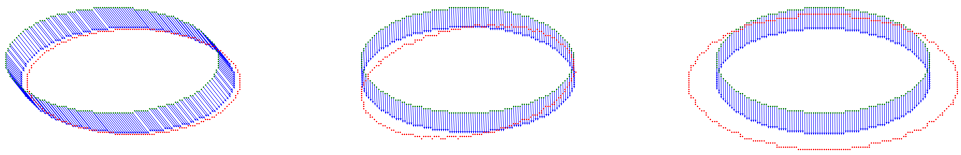
I demonstrate both two and six parameter models. The two parameter model is defined by Eq. 3.3, and the six parameter model is defined by Eq. 3.4. Coordinates of a point (x, y) in the first frame move to (x', y') in the next frame. The vector $\mathbf{w} = (w_1 \;\; w_2)^T$ is the vector of parameters for the two parameter model. The vector $\mathbf{w} = (w_1 \;\; w_2 \;\; w_3 \;\; w_4 \;\; w_5 \;\; w_6)^T$ is the vector of parameters for the six parameter model. The values of these parameters are estimated using the displacement vectors and a least squares approximation.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} + \begin{pmatrix} x \\ y \end{pmatrix} \tag{3.3}$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} w_1 \\ w_4 \end{pmatrix} + \begin{pmatrix} w_2 & w_3 \\ w_5 & w_6 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{3.4}$$

For the six parameter model, to obtain the displacement $\vec{u}(x,y) = (\delta x \;\; \delta y)^T$ of (x,y) we subtract $(x \;\; y)^T$ from both sides of (3.4). The left hand side of (3.4) is replaced by $(\delta x \;\; \delta y)^T$. On the right hand side, w_2 and w_6 get replaced by $w_2^1 = w_2 - 1$ and $w_6^1 = w_6 - 1$. The normal displacement field at (x,y) is given by $u_n(x,y) = \delta\vec{r}_n \cdot \vec{n} = n_x\delta x + n_y\delta y = w_1 n_x + w_2^1 x n_x + w_3 y n_x + w_4 n_y + w_5 x n_y + w_6^1 y n_y = \mathbf{w}\cdot\mathbf{p}$, where $\vec{n} = n_x\vec{\imath} + n_y\vec{\jmath}$ is the gradient direction, $\mathbf{p} = (n_x \;\; x n_x \;\; y n_x \;\; n_y \;\; x n_y \;\; y n_y)^T$, and $\mathbf{w} = (w_1 \;\; w_2^1 \;\; w_3 \;\; w_4 \;\; w_5 \;\; w_6^1)^T$ is the vector of parameters.

For each edge point $\vec{r}_i$ we have one normal flow value $u_{n,i}$, that we use as an estimate of the normal displacement at the point, a vector $\mathbf{p}_i$ computed from (x_i, y_i) and $\vec{n}_i$, and an approximate equation $\mathbf{w}\cdot\mathbf{p}_i \approx u_{n,i}$. Let the number of edge points be $N \geq 6$. We need to find a solution of $P\mathbf{w} - \mathbf{b} = \mathbf{e}$, where $\mathbf{b}$ is an N-element vector with elements $u_{n,i}$, P is an $N \times 6$ parameter matrix with rows $\mathbf{p}_i$, and $\mathbf{e}$ is an N-element error vector. We seek the affine model $\mathbf{w}$ that minimizes $\|\mathbf{e}\| = \|\mathbf{b} - P\mathbf{w}\|$; the solution satisfies the system $P^T P w = P^T\mathbf{b}$ and corresponds to the linear least squares (LS) solution. A similar methodology is used for estimating the parameters of the two parameter model. Once estimated, the affine model can be reprojected into vector form.

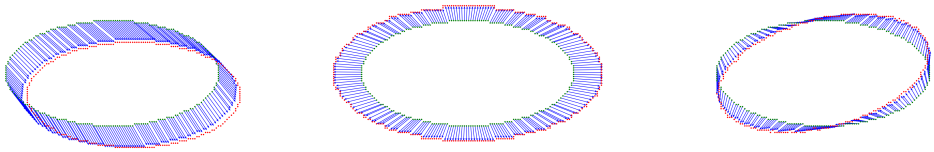


(a) rot:0 scale:1 x:10 y:10 (b) rot:10 scale:1 x:0 y:10 (c) rot:0 scale:1.25 x:0 y:10

Figure 3.11: Displacements computed using the proposed method and a two parameter motion model.

Figure 3.11 shows the reprojected vectors for a two parameter motion model. Notice

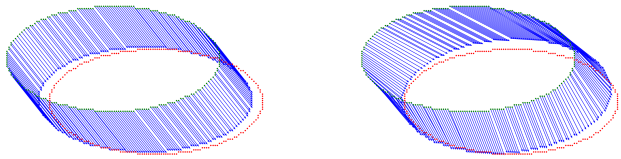
the vectors for 3.11b and 3.11c do not capture rotation or scale of the shape. This is a limitation of the two parameter model. For the translation-only movement in 3.11a, the vectors correspond to true motion.



(a) rot:0 scale:1 x:10 y:10 (b) rot:0 scale:1.25 x:0 y:0 (c) rot:15 scale:1 x:0 y:0

Figure 3.12: Displacements computed using the proposed method and a six parameter motion model.

Figure 3.12 shows the reprojected vectors for a six parameter motion model. These vectors correspond to the true motion of the shapes. Rotation and scale are captured by the model.



(a) two parameter model (b) six parameter model

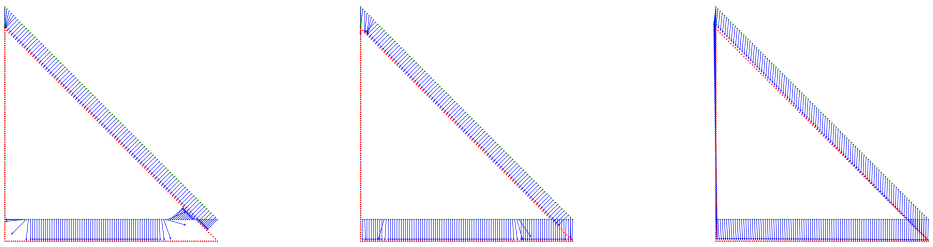
Figure 3.13: Comparison between two and six parameter models for rot:0 scale:1 x:20 y:20

Figure 3.13 shows a comparison between the two parameter and the six parameter models for a translation-only motion. As is evident from the figure, the two parameter

model is a better estimate of true motion. However, both models have some degree of error. This error can be quantified by comparing reprojected vectors from the estimated affine model to the reprojected vectors from the true affine model. Since the motions of the shapes are synthetic, the true affine model is available. For n motion vectors from the estimated affine model e and n motion vectors from the true affine model t, the average error is defined by

$$E(e,t) = \frac{1}{n}\sum_{1}^{i} d(e_i, t_i)$$

where $d(p,q)$ is the Euclidean distance, i.e. $d(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$. For the displacements in Figure 3.13, the average error for the two parameter model is 4.56 and the average error for the six parameter model is 8.60.



(a) basic (b) contour normal corrected (c) six parameter affine

Figure 3.14: Comparison of basic, contour normal corrected, and affine displacements for rot:0 scale:1 x:0 y:10

Figure 3.14 shows a comparison of the proposed approach, the proposed approach with contour normals, and the proposed approach with affine modeling. The affine modeling results in vectors that most closely correspond to the true motion.

Chapter 4: Gait Recognition

The goal of gait recognition is to identify people in videos based on their unique walking pattern. Gait has several advantages compared to other biometrics. Unlike fingerprint or iris biometrics, gait does not require participation from the subject, and unlike face recognition, gait can be analyzed in low resolution videos captured from different viewpoints [21]. One downside is that a person's gait can vary depending on clothing and environmental conditions [23]. Regardless, gait recognition has many applications in security and surveillance, and it has received increased attention in recent years [21].

Gait recognition typically consists of subject detection, silhouette extraction, feature extraction, feature selection, and classification. While subject detection and silhouette extraction are important aspects in a real gait recognition system, many gait recognition databases have already completed these steps. Figure 4.1 shows a sample silhouette sequence from the database used in this chapter. These silhouettes have been size-normalized and stabilized on the torso.

Figure 4.1: Sample silhouette sequence from OU-ISIR Large Population Dataset [33].

A silhouette sequence is a time series composed of at least several frames. Single frames represent static information such as shape and pose, while sequences capture dynamical information about the motion of the silhouettes. Gait recognition methods use sequences

as input, and many methods, including the one I propose, assume that the sequences contain one full gait cycle. Given a sequence, which represents an individual's gait, the objective of gait recognition methods is to label the sequence with the correct identity.

It is important to make a distinction between frame-level features and sequence-level features. Frame-level features are associated with a single frame. Motion computed between a pair of frames is a frame-level feature. Sequence-level features are associated with the entire sequence. A sequence level feature might be an ordered collection of frame-level features. In this case, sequences could be compared using a time series analysis technique, such as dynamic time warping [34]. Sequence-level features might alternatively be some combination of frame-level features. For example, the Gait Energy Image is a sequence-level feature obtained by averaging frame-level features into a single representation. Sequences using this representation could then compared with Euclidean distance.

Both shape and motion are important factors in gait recognition. Some approaches use motion information implicitly in the form of spatiotemporal representations, such as those derived from the Motion Energy Image [23, 24]. There are also some approaches that use motion information explicitly by computing traditional optical flow on silhouette images [29, 31].

I propose two approaches to gait recognition using displacements computed with the method described in Chapter 3. Unlike traditional optical flow techniques, my method is specifically designed to work on object contours, which are easily obtainable from silhouette images. Additionally, my method is faster than traditional optical flow techniques, supporting real-time gait recognition systems. I test my approaches on the OU-ISIR Large Population Dataset [33] and show that I achieve rank 1 and rank 5 performance that is comparable to the state-of-the-art.

4.1 Method

The dataset used in this experiment contains one gallery and one probe sequence per subject. Gallery sequences are analogous to a training set while probe sequences are analogous to a

testing set. The classifier can use the gallery sequences to model an individual's gait and the probe sequences to test its models. Since there is one gallery and one probe sequence per subject, this naturally corresponds to a nearest neighbor classifier. In the nearest neighbor framework, features are extracted for all gallery and probe sequences. Any given probe sequence is then classified by computing the similarity between the probe sequence and every gallery sequence. The label assigned to the probe sequence is the same as the one belonging to the most similar (nearest) gallery sequence. Figure 4.2 shows an overview of the framework.

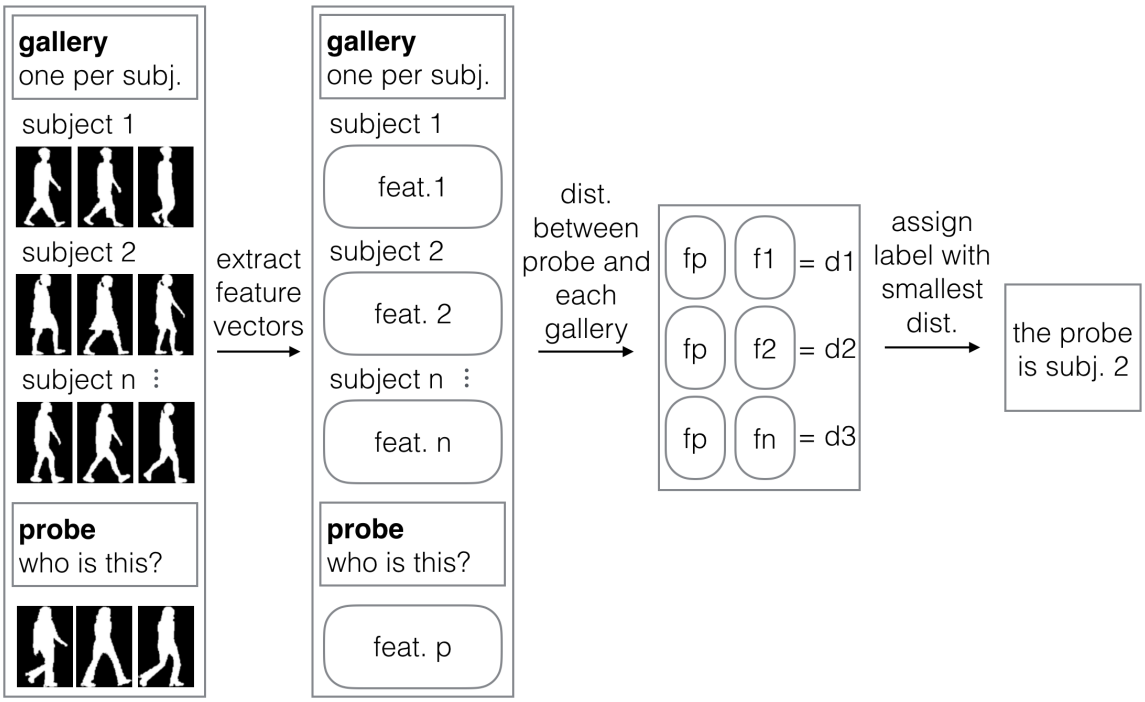


Figure 4.2: Overview of gait recognition framework.

In this framework, the key aspect is how to compute similarity between sequences: feature extraction, feature selection, and distance metrics. I propose an approach based on my method for computing displacements between contours. Contours are easily extracted from silhouette images by tracing edge points. Displacements are then computed between

contours in sequential pairs of frames and associated with the original frame. This process results in a motion vector, consisting of an x and y component, for each point along the contour in each frame. The last frame in the sequence is ignored because there is no successive frame from which to compute motion. These frame-level features are then averaged together to compute a sequence-level feature.

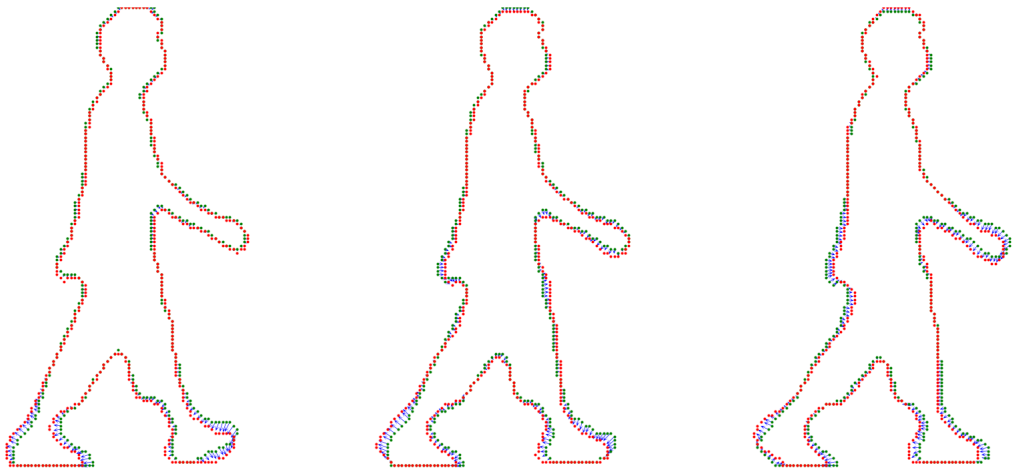


Figure 4.3: Displacements computed on frames from the silhouette sequence of Figure 4.1

Figure 4.3 shows displacements computed on frames from the silhouette sequence of Figure 4.1. I chose to use the method that factors in contour normals due to curvature of the silhouettes and the potential for large displacements when the individual moves quickly. Additionally, it is straightforward to compute contour normals using the silhouette image gradient. The threshold for contour normal difference was set to 30 degrees. The maximum vector distance was limited to 15. The affine modeling approach was not selected because the silhouette contour is not a rigid structure. Since the shapes of contours vary, they cannot be directly compared between frames. I designed two representations, the Histogram of Motion (HOM) and the Edge Motion Vector (EMV), that allow motion to be compared between frames.

4.1.1 Histogram of Motion

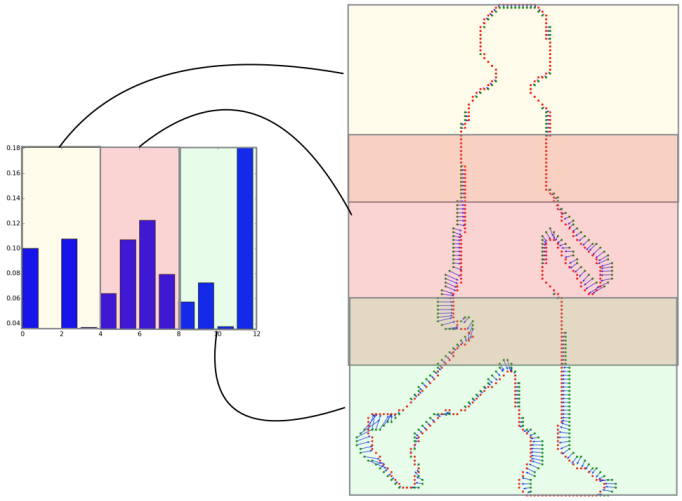


Figure 4.4: Overview of Histogram of Motion.

The Histogram of Motion (HOM) constructs histograms from motion vectors. Each vector in a given frame is converted from an x, y representation into a magnitude, direction representation. Magnitude is computed using

$$\sqrt{x^2 + y^2} \tag{4.1}$$

and direction is computed using

$$arctan2(y, x) \tag{4.2}$$

The image is divided into h evenly spaced, overlapping horizontal and v evenly spaced, overlapping vertical patches. The purpose of using multiple patches is to preserve spatial locality of the motions. A histogram with b bins is computed for each image patch. Each bin corresponds to a direction of motion, with directions ranging from 0 to 360 degrees. The bins are equal-width so that the entire range is divided evenly among all bins. Each bin contains the sum of the magnitudes of vectors pointing in the bin's direction. All of the

$h + v$ histograms are concatenated into a single feature vector of length $(h + v) * b$. I use $v = 7$ vertical patches, $h = 12$ horizontal patches, and $b = 16$ bins, resulting in a feature vector length of 304. The feature vector f is normalized to sum to 1, thereby converting it into a probability density function.

A histogram is computed for each frame of a sequence. Each frame-level histogram is then averaged to construct a sequence-level histogram. If a sequence contains l frames, then the corresponding Histogram of Motion is defined by

$$HOM = \frac{1}{l} \sum_{i=1}^{l} f_i \tag{4.3}$$

At the cost of losing information, this process simplifies the time series into a single representation per sequence. This is a common approach in literature [21], and as the results will show, discriminatory information is still preserved.

Sequences are compared using Bhattacharyya distance. For probability distributions p and q over the same domain X, Bhattacharyya distance is defined as

$$D_b(p, q) = -ln(BC(p, q)) \tag{4.4}$$

where

$$BC(p, q) = \sum_{x \in X} \sqrt{p(x)q(x)} \tag{4.5}$$

is the Bhattacharyya coefficient. This distance metric is designed specifically to measure distance between two probability distributions. Experimentation shows that this distance metric works better than euclidean distance in this application.

4.1.2 Edge Motion Vector

The Edge Motion Vector directly compares motion vectors on the contour by projecting the sides of the contour onto a line. Let h be the number of rows in the image and p and

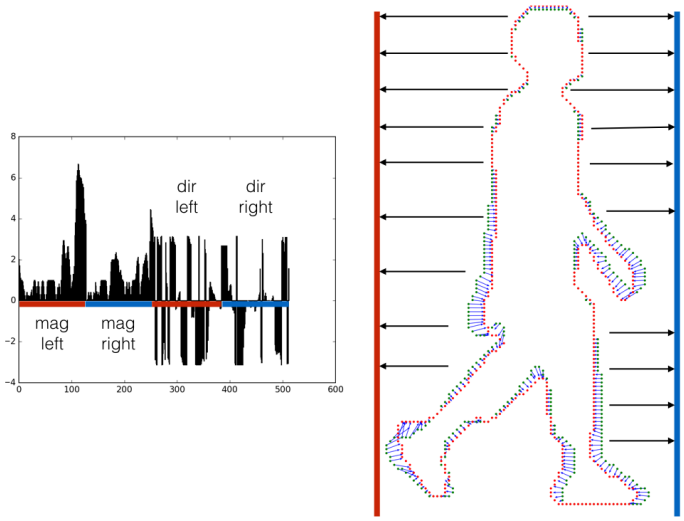


Figure 4.5: Overview of Edge Motion Vector.

r be vectors of length h. For each row in the image, the leftmost contour point is placed into the corresponding location in p, and the rightmost contour point is placed into the corresponding location in r. If there are no contour points in a given row, zero is placed into both p and r. If there is only one contour point in a given row, that point is placed into both p and r. Like the Histogram of Motion approach, the contour points in both p and r are converted from an x, y representation to a magnitude, direction representation using Eq. 4.1 and 4.2. The points in p and r are then arranged into a feature vector f of length $h * 4 = (l + r) * 2$. The first h spots in f are the magnitudes of the points in p. The second h spots in f are the magnitudes of the points in r. The third h spots in f are the directions of the points in p. The last h spots in f are the directions of the points in r. The images in the database are $h = 128$, making the feature vector length 512.

A feature vector is computed for each frame of a sequence. Each frame-level vector is then averaged to construct a sequence-level vector. If a sequence contains l frames, then

the corresponding Edge Motion Vector is defined by

$$EMV = \frac{1}{l}\sum_{i=1}^{l} f_i \tag{4.6}$$

Sequences are compared using euclidean distance between Edge Motion Vectors.

4.2 Experimental Design

I evaluated my approaches on the OU-ISIR Large Population Dataset (Subset A) [33]. The dataset consists of registered and size-normalized (128x88) silhouette sequences that have been stabilized on the torso. There are four viewing angles $v = 55$, $v = 65$, $v = 75$, and $v = 85$ degrees. There is a single gallery and a single probe sequence per viewing angle per subject. Table 4.1 shows the number of subjects in the database.

Table 4.1: The number of subjects for each viewing angle in the database.

v	**# subjects**
55	3714
65	3778
75	3759
85	3254

Performance was evaluated using the rank 1 and rank 5 evaluation metrics. This is a common evaluation metric in gait recognition literature. The rank list is a ranking of all gallery subjects in terms of distance to a given probe. Rank 1 corresponds to the percentage of correct subjects appearing in the first spot of the rank list. Rank 5 corresponds to the percentage of correct subjects appearing in any of the first five spots.

I ran several experiments to test my approaches. I compared the performance of my

approaches, Histogram of Motion and Edge Motion Vector, for each viewing angle $v = 55$, $v = 65$, $v = 75$, and $v = 85$ with $n = 300$ randomly selected subjects. I further tested the better performing representation, the Edge Motion Vector, for all subjects and all viewing angles and compared those results to the published results for gait energy image and gait flow image. I also measured the rank 1 and rank 5 performance of the Edge Motion Vector with varying numbers of subjects n. This revealed how the number of subjects affects performance. For a 75 degree viewing angle, I tested the Edge Motion Vector with $n = 10$, $n = 100$, $n = 200$, $n = 300$, $n = 700$, $n = 1000$, $n = 2000$, and $n = 3759$ subjects.

4.3 Results

Table 4.2 shows rank 1 and rank 5 performance of EMV and HOM for viewing angles $v = 55, 65, 75, 85$ and $n = 300$ subjects. EMV achieves better performance for all viewing angles. This result may be due to a variety of reasons including the fact that EMV directly compares magnitudes and directions of corresponding vectors rather than binning vector magnitudes like HOM.

Table 4.2: Performance of EMV and HOM for viewing angles $v = 55, 65, 75, 85$ and $n = 300$ subjects.

	Rank 1		**Rank 5**	
v	**EMV**	**HOM**	**EMV**	**HOM**
55	83	80	94	92
65	85	81	96	92
75	88	83	97	95
85	88	80	96	92

Table 4.3 shows rank 1 and rank 5 performance of EMV, Gait Energy Image (GEI), and Gait Flow Image (GFI), as reported by [33]. EMV achieves better performance than

GFI, which is an approach that also computes motion explicitly. Compared to GEI, EMV performs 4.9-8.3 percent worse for rank 1 and 2.5-3.9 percent worse for rank 5.

Table 4.3: Performance of EMV, Gait Energy Image (GEI), and Gait Flow Image (GFI) for $v = 55, 65, 75, 85$ and $n = max$ subjects.

	Rank 1			**Rank 5**		
v	**EMV**	**GEI**	**GFI**	**EMV**	**GEI**	**GFI**
55	77.6	84.7	75.2	88.5	92.4	85.8
65	78.3	86.6	77.1	89.0	92.8	87.3
75	80.3	86.9	76.5	90.3	92.8	85.8
85	80.8	85.7	74.9	90.1	93.0	84.7

Figure 4.6 shows rank 1 and rank 5 performance of EMV for $v = 75$ degrees and $n = 10$, $n = 100$, $n = 200$, $n = 300$, $n = 700$, $n = 1000$, $n = 2000$, and $n = 3759$ subjects. The rank 5 performance decreases steadily as the number of subjects is increased. This result reflects the fact that the more subjects in the gallery, the more likely it is that there will be subjects with similar gait profiles. Rank 1 performance also decreases as the number of subjects is increased, but it is more volatile than the smooth decrease in rank 5 performance. This difference is due to the nature of rank 1 and rank 5. For a successful rank 5 classification, the true label can appear in any of the first 5 spots of the rank list. The number of subjects with similar gait profiles required to make a rank 5 classification unsuccessful is more than the number required to make a rank 1 classification unsuccessful. Additionally, the large drop in rank 1 performance for the experiment with $n = 2000$ subjects can be explained by the randomization of subjects for each experiment. The subjects used in the $n = 2000$ experiment contained proportionally more subjects that were hard to discriminate (that is, subjects whose gait profiles are similar).

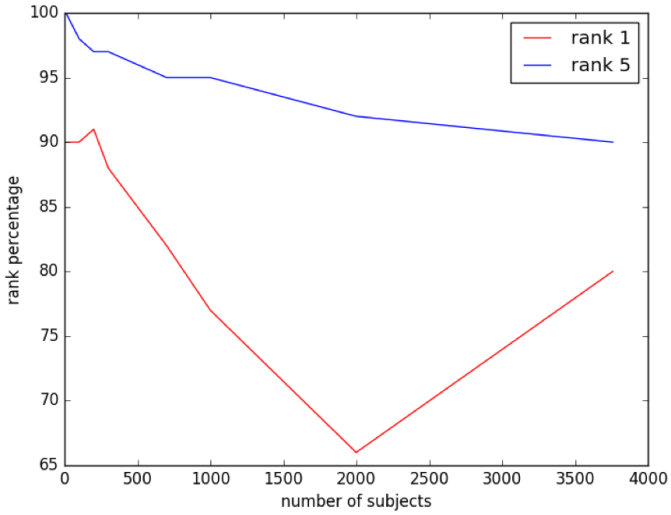


Figure 4.6: Performance of EMV for $v = 75$ degrees and $n = 10$, $n = 100$, $n = 200$, $n = 300$, $n = 700$, $n = 1000$, $n = 2000$, and $n = 3759$ subjects.

Chapter 5: Conclusion and Future Work

In this thesis, I presented a new method for estimating motions of object contours, and I demonstrated the successful use of the method for gait recognition. The basic method uses the distance transform to establish correspondences between point sets. The resulting displacement vectors are mostly consistent with normal flow. I carefully analyzed the method's output and revealed how the medial axis and curvature of the object, as well as the magnitude of motion, affect the computation of displacement vectors. This analysis led to an extension of the method where contour orientation was used to correct some displacements. Affine modeling was further used to compute the true motions of objects. After describing the method in detail, I applied the method to the practical problem of gait recognition. The method worked successfully on low resolution contours obtained from silhouette images. I designed two representations, the Histogram of Motion and the Edge Motion Vector, to compare features between frames and sequences. These approaches achieved promising results. The successful application of the method to gait recognition proves that the method's output is useful as a standalone representation.

The method has implications beyond being a standalone representation. Other techniques for estimating motion as well as other application domains could benefit from the proposed method. Optical flow techniques have used a coarse-to-fine estimation to speed up computation. The proposed method could serve as a seed to such techniques. Chamfer matching could also be sped up using the proposed method. An additional application domain is action recognition, where preliminary work using the output of the proposed method as input into a convolutional neural network has yielded encouraging results.

Outside of application to other domains, the method itself could benefit from several improvements. Although the focus in this thesis has been on objects, contours can be obtained from a variety of sources including image edges. As long as edges correspond to

stable regions in an image, this modification could yield a denser motion field. Currently, the motion modeling approach only works on rigid objects. This could be extended by applying motion modeling to small patches in the image where a loose assumption of rigidity could be made. As for gait recognition, a significant improvement could be made by applying a time series analysis technique such as dynamic time warping rather than averaging frame-level features. Such an approach would preserve sequential information. An approach using key frames could provide a medium between dynamic time warping and averaging all frame-level features. Furthermore, using more features such as shape features, could improve performance.

Bibliography

Bibliography

[1] B. K. Horn and B. G. Schunck, "Determining optical flow," in *1981 Technical symposium east*. International Society for Optics and Photonics, 1981, pp. 319–331.

[2] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision." in *IJCAI*, vol. 81, 1981, pp. 674–679.

[3] M. J. Black and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields," *Computer vision and image understanding*, vol. 63, no. 1, pp. 75–104, 1996.

[4] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision*. Springer, 2004, pp. 25–36.

[5] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2432–2439.

[6] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*. IEEE, 1994, pp. 593–600.

[7] T. Brox, C. Bregler, and J. Malik, "Large displacement optical flow," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 41–48.

[8] L. S. Davis, Z. Wu, and H. Sun, "Contour-based motion estimation," *Computer Vision, Graphics, and Image Processing*, vol. 23, no. 3, pp. 313–326, 1983.

[9] J. S. Park and J. H. Han, "Estimating optical flow by tracking contours," *Pattern recognition letters*, vol. 18, no. 7, pp. 641–648, 1997.

[10] ——, "Contour motion estimation from image sequences using curvature information," *Pattern recognition*, vol. 31, no. 1, pp. 31–39, 1998.

[11] ——, "Contour matching: a curvature-based approach," *Image and Vision Computing*, vol. 16, no. 3, pp. 181–189, 1998.

[12] J. H. Han and J. S. Park, "Contour matching using epipolar geometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 4, pp. 358–370, 2000.

[13] C. Liu, W. T. Freeman, and E. H. Adelson, "Analysis of contour motions," in *Advances in Neural Information Processing Systems*, 2006, pp. 913–920.

[14] K. Soeder, "Estimating motion of object contours from distance transforms," Ph.D. dissertation, George Mason University, 2015.

[15] A. Rosenfeld and J. L. Pfaltz, "Distance functions on digital pictures," *Pattern recognition*, vol. 1, no. 1, pp. 33–61, 1968.

[16] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno, "2d euclidean distance transform algorithms: A comparative survey," *ACM Computing Surveys (CSUR)*, vol. 40, no. 1, p. 2, 2008.

[17] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," Cornell Computing and Information Science, Tech. Rep., 2004.

[18] G. Borgefors, "Hierarchical chamfer matching: A parametric edge matching algorithm," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 10, no. 6, pp. 849–865, 1988.

[19] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the hausdorff distance," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 9, pp. 850–863, 1993.

[20] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21, no. 13, pp. 1145–1153, 2003.

[21] J. Wang, M. She, S. Nahavandi, and A. Kouzani, "A review of vision-based gait recognition methods for human identification," in *Digital Image Computing: Techniques and Applications (DICTA), 2010 International Conference on*, Dec 2010, pp. 320–327.

[22] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

[23] J. Man and B. Bhanu, "Individual recognition using gait energy image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 2, pp. 316–322, Feb 2006.

[24] J. Liu and N. Zheng, "Gait history image: A novel temporal template for gait recognition," in *Multimedia and Expo, 2007 IEEE International Conference on*, July 2007, pp. 663–666.

[25] C. Chen, J. Liang, H. Zhao, H. Hu, and J. Tian, "Frame difference energy image for gait recognition with incomplete silhouettes," *Pattern Recognition Letters*, vol. 30, no. 11, pp. 977–984, 2009.

[26] K. Bashir, T. Xiang, and S. Gong, "Gait recognition using gait entropy image," in *Crime Detection and Prevention (ICDP 2009), 3rd International Conference on*. IET, 2009, pp. 1–6.

[27] E. Zhang, Y. Zhao, and W. Xiong, "Active energy image plus 2dlpp for gait recognition," *Signal Processing*, vol. 90, no. 7, pp. 2295–2302, 2010.

[28] C. Wang, J. Zhang, J. Pu, X. Yuan, and L. Wang, "Chrono-gait image: a novel temporal template for gait recognition," in *European Conference on Computer Vision*. Springer, 2010, pp. 257–270.

[29] K. Bashir, T. Xiang, and S. Gong, "Gait representation using flow fields," in *British Machine Vision Conference, BMVC 2009, London, UK, September 7-10, 2009. Proceedings*, 2009, pp. 1–11. [Online]. Available: http://dx.doi.org/10.5244/C.23.113

[30] T. H. Lam, K. Cheung, and J. N. Liu, "Gait flow image: A silhouette-based gait representation for human identification," *Pattern Recognition*, vol. 44, no. 4, pp. 973 – 987, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320310004954

[31] Y. Yang, D. Tu, and G. Li, "Gait recognition using flow histogram energy image," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*, Aug 2014, pp. 444–449.

[32] T. C. Hales, "The jordan curve theorem, formally and informally," *American Mathematical Monthly*, vol. 114, no. 10, pp. 882–894, 2007.

[33] H. Iwama, M. Okumura, Y. Makihara, and Y. Yagi, "The ou-isir gait database comprising the large population dataset and performance evaluation of gait recognition," *IEEE Trans. on Information Forensics and Security*, vol. 7, Issue 5, pp. 1511–1521, Oct. 2012.

[34] N. V. Boulgouris, K. N. Plataniotis, and D. Hatzinakos, "Gait recognition using dynamic time warping," in *Multimedia Signal Processing, 2004 IEEE 6th Workshop on*, Sept 2004, pp. 263–266.

Curriculum Vitae

Sam Gelman received his Bachelor of Science in Computer Science from George Mason University in 2014. His research interests include computer vision and machine learning.