JOINT TARGET TRACKING AND RECOGNITION

USING SHAPE-BASED GENERATIVE MODEL

By

LIANGJIANG YU

Bachelor of Science in Communication Engineering

North University of China

Taiyuan, Shanxi, China

2008

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2011

JOINT TARGET TRACKING AND RECOGNITION

USING SHAPE-BASED GENERATIVE MODEL

Thesis Approved:

Dr. Guoliang Fan

Thesis Adviser

Dr. Martin Hagan

Dr. Damon Chandler

Dr. Nazanin Rahnavard

Dr. Mark E. Payton

Dean of the Graduate College

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

CHAPTER I

INTRODUCTION

Automated target tracking and recognition (ATR) is the operating system that no human being is involved in, but only under its own supervision. It is based on the transmission of many kinds of signals processed by computers according to a specified program (Figure 1.1). It can automatically detect and lock targets that appear in the data and also recognize its class. The ATR can return us the 3D coordinate or location, and also the pose information of the target, and also the label can be presented. No human get involved in this process so that the results are quite stable and reliable.

Figure 1.1: Example of ATR application.

The human eyes can learn objects in a scene easily and also recognize individual or multi-objects quickly. So that replicating this ability at least partially is the main purpose and challenging issue that triggered our research in the area of object recognition. There are two challenge issues: the first one is about models that can capture the soul of a class or category; the other one is how the metrics are defined and how we can better match our models to data effectively, so that we can do an efficient reference.

Some of those research focus on human or car detection [1, 2], while some others try to tell the difference between multiple classes [3]. And also some key issues need to be considered such as how to detect and recognize objects in a scene and how to account for both inter-class and intra-class variations. Usually both the identity and view factors are treated as discrete values [4-6], while in this paper we introduce an algorithm that both of these variables are continuous, which makes it much more flexible to deal with unknown views or class variants. For the appearance model, the major challenge in the vision field is the target appearance variations based on different viewports and underlying 3D structures. Moreover, the identity is usually represented by discrete variables in most ATR algorithms. So the method we will introduce is using this generative model to deal with both of these two factors in a continuous manner, coupling both view and identity manifold for target representation, and it will significantly make the ATR inference process easier, allowing us to meaningfully analyze new target appearances and hence to handle unknown targets under unknown views or previous viewpoints that cannot be seen.

In nature target representations in the IR data are usually non-parametric, including templates, histograms, edge features and so on [6-8]. In [7] the target is characterized by shape features, intensity and a self-organizing map is used for classification. Histogram-based representations [8] have been shown to be simple and robust under difficult tracking conditions [9], but such methods cannot distinguish effectively between various target classes because they do not have enough shape information. In [10], the shapes and poses variability is represented by deformable models with many parameters that have to be optimized later for localization and recognition. Moreover, this method has an assumption that, observations have enough resolution and the boundary of targets can be well detected and evaluated against the proposed parametric model. Many existing IR target recognition approaches [6, 11, 12] depend on the use of exemplar templates from several viewpoints to train a classifier that can associate the observed target appearance with a discrete identity variable. Such methods usually need a very dense training set

of viewpoints to successfully recognize targets from arbitrary views and they are often not capable of handling unknown targets that did not appear in the training data.

Particularly we introduce a shape/silhouettes based generative target model developed by Dr. Vijay, which is controlled by both continuous view and identity variables on two manifolds for multi-view target modeling. The simpe view of this generative model is shown in Figure 4.2. the identity manifold is a 1D circle structure which is controlled by only one variable, and the view manifold is a 2D hemisphere which contains two variables: elevation (φ) and azimuth (θ) angle. A non-linear tensor decomposition technique is used to couple the two manifolds into a compact generative model which can be incorporated in a particle filter based inference algorithm for ATR purpose. The most convenience of this model is the controlling of only three variables. Our research in this paper is based on the new ATR database released by the Military Sensing Information Analysis Center (SENSIAC) (Figure 1.2) [13] which contains a large amount of infrared imagery dataset of many different military and civilian vehicles: BTR70 Armored Personnel Carrier (APC), T72 Main Battle Tank, BRDM2 Infantry Scout Vehicle, BMP2 APC, ZSU23 Tank with Anti-Aircraft Weapon, 2S3 Tank with Self-propelled Howitzer, MTLB Armored Reconnaissance Vehicle Towing a D20 Artillery Piece, Ford Pick-up and ISUZU SUV. Particularly, except from the MTLB with an artillery piece, we use the rest of these vehicle's data in our experiments, and group them into four different classes. (1) Tanks: T72, 2S3, and ZSU23; (2) APCs: BTR70, BRDM2 and BMP2; (3) SUVs: ISUZU SUV and (4) Pick-ups: Ford Pickup. Moreover, we will use two visible video data taken in an indoor lab of two remote controlled toy vehicles: Lexus IS350 sedan, BMW X5 SUV (Figure 1.3 (a) and (b)). And also we collected another visible data from PETS2001[14] of a white cargo van (Figure 1.3 (c)). So totally we have six classes: Tanks, APCs, SUVs, Pick-ups, Cars, and Mini-vans.

To evaluate the efficiency of the generative target model, we designed four different methods to handle the view and identity factors, and we will also define a set of quantitative metrics for

Figure 1.2: IR imagery from SENSIAC[13] (Left: BTR70 APC; Right: T72 Main Battle Tank).

(a) (b) (c)

Figure 1.3: Visible-band imagery ((a) and (b): videos taken in VCIPL of RC toy vehicles Lexus IS350 sedan car and BMW X5 SUV respectively, and (c): video from PETS2001 [14]).

algorithm evaluation. Experimental results shows that the generative model can improve the tracking and recognition accuracy significantly, and the advantages of dealing with both identity manifold and view manifold for target tracking and recognition is demonstrated as well, both qualitatively and quantitatively.

This thesis is organized in the order that the research was completed. First we will talk about the related work in Chapter 2, introducing previous work that has been done in the field of motion

and appearance model, and also ATR algorithm. Then we will begin talking about the feature extraction method and camera calibration in Chapter 3, mainly focusing on background subtraction through which we can get the initial silhouettes that we need for the shape based ATR algorithm. The camera calibration is important since during our inference process we need to compare the position hypothesis with the true data using camera projection matrix. Then in Chapter 4, we will introduce our view and identity manifold and hence the tensor decomposition method which couple these two manifolds into one generative manifold. And most importantly we will talk about the learning of this generative model, and also the reconstruction of target silhouettes theoretically.

Then in Chapter 5, we will focus on our inference algorithm, using particle filter based target tracking and recognition method that applies the recently developed target model for ATR purpose testing. In details, we will talk about the motion model we use for the inference process, and also the training appearance we need for the learning of the generative model, or how do we get the target silhouettes for the training.

Then finally, in Chapter 6, we will concentrate on our experimental processing, showing the learning process of the generative model, and several experiments of how do we interpolate target shape between training viewpoints, and also the interpolation between and within classes. Then, we will apply the recently proposed generative model into ATR applications from SENSIAC dataset and some visible video data, by comparing four different methods we developed in which the ways of handling target view and identity are different. We will also mention the limitations and possible extensions of our algorithm. Finally we will present the conclusions in Chapter 7.

CHAPTER II

RELATED WORK

In this chapter we will talk about several related work that has been done previously related to different ways of representing 3D objects, and then we will examine their capabilities to parameterize shape variations, and the abilities of interpolation to make the parameter estimation easier.

2.1 Multi-view target models

Object representation has two branches of ideas, one is about a set of 2D snapshots [15, 16], and the other one is about the 3D object models [17]. For the first one, we can interpolate unknown views through given ones, and for the second one, the 3D model is usually used to match with the 2D observation using 3D to 2D projection, mainly using the 3D-2D projection camera matrix obtained by camera calibration. And similarly, object recognition issues could also be grouped into two different categories: the first one is those involving 2D multi-view images [18-23], and the second one is those supported by explicit 3D models [24-27]. Some researches try to use both of the 3D shape and 2D information. Various 2D features such as silhouettes, edges, HOG, and SIFT, or 3D models such as meshes, polygons, and polyhedrons were used in these methods. But the psychophysical experiments guide us to use 2D view-based silhouettes for multi-view object representation. There are also hybrid methods [3] that is using both of the 3D shapes and 2D information.

2.1.1 Templates Methods

The Simplest multi-view model is the 2D Templates. These are representative view samples of many different targets in many different views. Some examples will be shown in Figure 2.1. In this work, they use two specific photographs and define an adjustable skeleton of several points, the structure of skeleton compatible formats. They define two template models, male and female, which are naked [28].

Figure 2.1: 3D human shape reconstruction using template model [28].

The limitation of this kind of models is that we have to save a large number of training model clips, which is quite challenging and not efficient.

2.1.2 3D Models Approaches

The 3D multi-view models are the 3D representation of target shape, and it can be of two types. (1) The first one is rigid models (Figure 2.2), these are complete mesh or vertices based models. One previous work [25] is proposed to use this kind of model, aims at tracking vehicles from monocular intensity image sequences and presents an efficient and robust approach to 3D model based vehicle tracking. Although it is very accurate, it is also difficult to store, and we need to project the 3D model into 2D at each frame to perform inference.

(2) The second one is the parametric model, in which the target shape is roughly predefined, and several segments are joined together to form a shape. An example is showing how this model looks like (Figure 2.3). In [10] a deformable 3D geometric vehicle model with 12 parameters is set up as prior information and Bayesian Classification Error is adopted for evaluation of fitness between the model and images. Using a novel evolutionary computing method called EDA (Estimation of Distribution Algorithm), we can not only determine the 3D pose of the vehicle but also obtain a 12 dimensional vector which corresponds to the 12 shape parameters of the model [10]. But these models need optimization at each frame to determine the correct set of parameters, and also there is no clear distinction between target types.

Figure 2.2: 3D rigid model example.

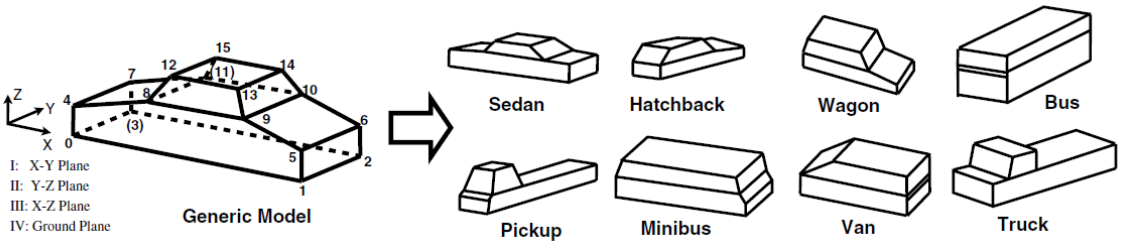


Figure 2.3: A generic parametric 3D vehicle model which can be deformed to fit with different vehicles [10].

(3) The third one is using the 3D feature model. This method is based on a new 3D feature model. Instead of using a complicated mechanism for relating multiple 2D training views, this method establishes spatial connections between these views by mapping them directly to the surface of 3D model [29] (Figure 2.4). Still, this method needs to store a large amount of training 3D feature model images, and the computational load is quite large.

Figure 2.4: Construction of 3D feature model for motorbikes [29].

2.2 Generative Models

Two topics are related to each other on shape representation. The first one is how we can characterize the variation of shapes, and the second one is how to effectively infer the hidden variables of shape, including the viewpoints and identity information [30]. Feature vectors obtained from common shape descriptors like shape contexts [31], moment descriptors [32] etc. are usually assumed to be in a Euclidean space to ease shape modeling and recognition. But in most cases, we can learn the underlying shape better by some non-linear dimension reduction methods through non-linear manifold in a lower dimension space. However these kinds of

manifold learning are usually either view dependent or identity dependent [33]. An identity independent view manifold was proposed in [34] where the identity variable is still discrete. Still, the lacking of a global shape representation framework that is supported by both view and identity variables in a LD space triggers our research. Another recent research trend is trying to find a space where every point in this space stand for a plausible shape and a curve between two points in this space stands for a deformation path between two shapes. This was shown effectively in action recognition [30] and shape clustering [35], it is still hard to separate the identity and view factors explicitly which is very necessary in ATR applications.

All these triggers us the research of learning the low dimension (LD) latent factors, view and identity, from the high dimension (HD) observations, or in other words, silhouettes. In an early work, [36], PCA method was used to find two separate eigen-spaces in order to do the learning of 3D objects, one for the identity and the other one for the pose. by decomposing observations into a bunch of independent factors, the bilinear models [37] and multi-linear analysis [38] provide a much more systematic multi-factor representation. In [39], a shape sub-manifold was introduced that can link the shape variable to the appearance, and we have to learn this for each category. All these methods introduced above have a limitation of discrete identity variable, where each object is associated to different view manifold.

2.2.1 PCA Approaches

Principal component analysis (PCA) is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values or uncorrelated variables called principal components, which are guaranteed to be independent only if the data set is jointly normally distributed (Figure 2.5). However, PCA is sensitive to the relative scaling of the original variables.

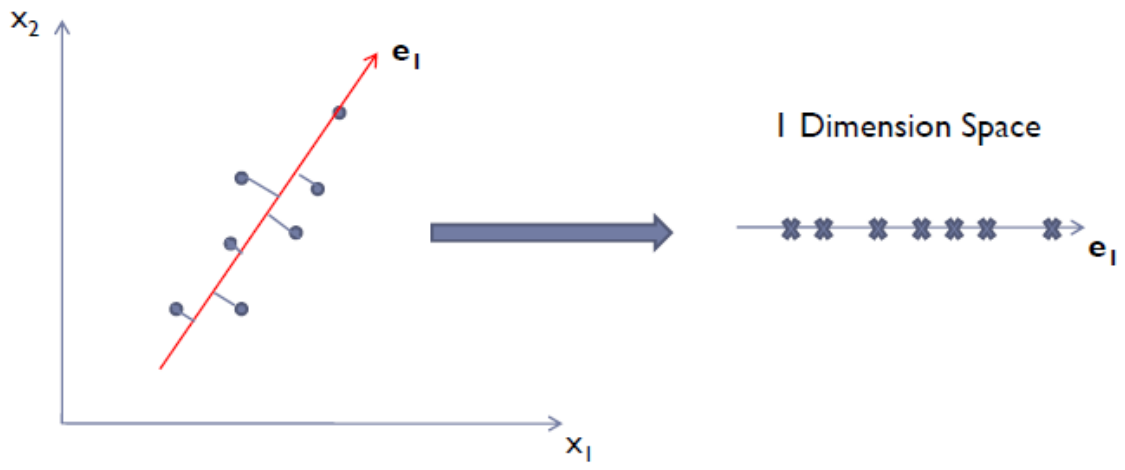


Figure 2.5: PCA: determine a subspace with dimension that captures the largest variation [36].

2.2.2 Multi-linear Tensor Analysis

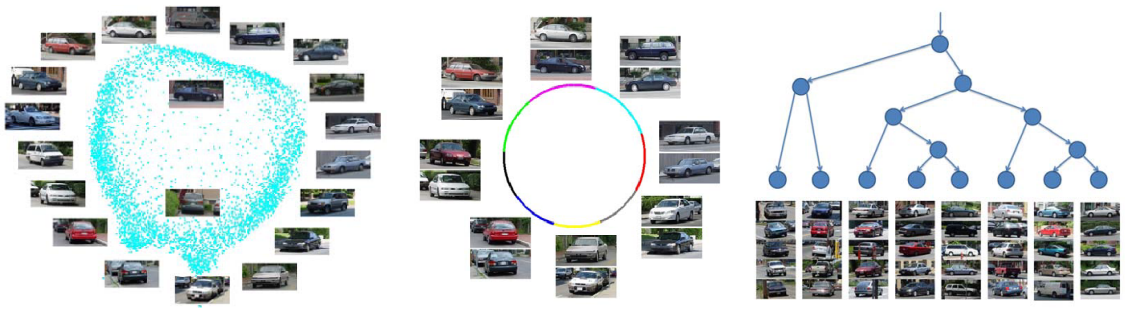
Another approach introduced in [38] using a multi-linear modeling technique, which employs a tensor extension of the conventional matrix singular value decomposition, known as the N-mode SVD, which is shown in Figure 2.6.

The limitation for both the linear PCA and multi-linear tensor approach is that there is no interpolation ability between training data. This means we still need to store a huge amount of data for training, and it is impossible to know and hence reconstruct data between training data.

2.2.3 Manifold Representation

One commonly used non-linear model is called manifold, which can identify low dimensional embedding using locally linear embedding (LLE), etc., such as discussed in [40], they adopt LLE to represent HD car samples in a 2D space and further normalize these 2D points onto a circle. Within this compact space, car samples are grouped into several sub-categories by the K-means clustering method, and then an agglomerative clustering method is used to construct a tree from top to down (Figure 2.7). But in this method, different targets lie in different subspaces, and they

Figure 2.6: Image set for human face. Left to right, different illuminations; within each of the three panels, expressions and views changes horizontally and vertically[38].



(a) (b) (c)

Figure 2.7: Multi-view car detection: (a) 2D embedding of cars using LLE; (b) normalized 2D points from (a) onto a circle, and then K-mean clustering is applied; (c) hierarchical clustering [40].

need to be aligned to be useful in the multi-target case, and there is no smooth change between each training targets, or no interpolation is available for intra-class identification.

2.3 Our Approach and Advantages

In this work, our target is represented by its representative 2D views, and we have two main reasons to do this. First, theoretically this is supported by the psychophysical views presented in [41] which suggests that *the human visual system is better described as recognizing objects by 2D view interpolation than by alignment or other methods that rely on object-centered 3D models.* Secondly, storing and using lots of detailed and fine 3D models of various target types in an ATR system can be very difficult and troublesome. Moreover, robust features such as HOG, SIFT that is being used to characterize objects in 2D views are all developed for visible images, and they are also limited by several issues such as the resolution of the image, and the quality, too. In IR dataset, targets are usually pretty small and we do not have enough resolution that can support the using of features. Specially, IR sensors in our dataset (SENSIAC) are static, which makes the preprocessing of the segmentation easier, using background –subtraction. Since it is possible to extract target shape information and also easier to represent the shape using silhouette, it motivates us to use this 2D view based silhouettes for multi-view target representation.

And we are also inspired much from the non-linear tensor decomposition proposed in [34] which a radial basis function (RBF) non-linear mapping was involved before multi-factor tensor analysis. In detail, both the view and identity variables are involved by an identity-independent view manifold shared by all target types and also a view-independent identity manifold. By this generative model, we can interpolate new shapes along both view and identity manifolds, and this is controlled by only three variables, which makes our inference process much more efficient and flexible.

With using the RBF kernel mapping and the non-linear tensor decomposition, we can factorize out the identity factor from view information, and hence build two independent manifolds: identity and view manifold. Most importantly, we don't need to store all the view information for each target, and we can either interpolate between each training view point, and also do inter-class and intra-class interpolation.

To illustrate the efficiency of this generative model, we develop a particle filter based ATR inference algorithm, which can estimate the 3D location, pose and the identity of a target appears in the scene. We also include more training dataset, not only civilian but also military vehicles for the learning of this generative mode, and put them all together on an identity manifolds, so that we can not only track and recognize civilian targets but also military targets. Experimental results of the interpolation shows smooth changes on both of the view manifold, and the identity manifold. This recently proposed generative model based particle filter inference algorithm is tested on both IR data and visible data, and the tracking and recognition results from overall 48 IR data sequences and 3 visible data sequences show the efficiency of the generative model for ATR purposes.

CHAPTER III

FEATURE EXTRACTION AND CAMERA CALIBRATION

In this chapter we will talk about the feature extraction and camera calibration, which are important for the future application of the generative model into IR data and visible-band data. Through feature extraction we can build the appearance model which can be used for the hypothesis evaluation during inference process, and camera calibration can help us to find the corresponding projected 2D point in an image from a given 3D point in the world coordinate system, which is also important during the inference process and the result evaluation.

3.1 Background Subtraction

Background modeling is usually used to detect the moving foreground (object) in various applications in a scene like video surveillance [42, 43], optical motion capture and multimedia [44, 45]. It involves the comparison between an observed image and an estimated static background image without objects of interest. Lots of work has been done related to this issue, and they can be classified as: Basis Background Modeling, Statistical Background Modeling, Fuzzy Background Modeling, and Background Estimation. It is well known that the most used one method is the Mixture of Gaussian (MOG) model, introduced by Stauffer and Grimson [46].

The idea of the MOG is that, each pixel is characterized by a Gaussian mixture model, then the probability of the observed pixel value is given by [47]:

$$P(x_i) = \sum_{i=1}^{K} \omega_{i,t} \cdot \eta(X_t, \mu_{i,t}, \Sigma_{i,t}), \tag{3.1}$$

where K is the number of distributions, or components in the MOG, and $\omega_{i,t}$ is i[th] weight associated with each component of the MOG, with mean of $\mu_{i,t}$ and a standard deviation $\Sigma_{i,t}$. Due to the computational load, here we assume that the RGB components are independent to each other. Actually in our SENSIAC dataset, infrared frames are all one channel instead of RGB. Experimentally, this is much easier. But for the visible data we are using, we will apply this assumption.

Then, we will use the K-mean method to initialize the weight. When a new frame is given, we can update the Gaussian distribution of each pixel. If the Mahalanobis distance from the sample to the component is less than a predefined constant, the sample is "close" to this component. Our algorithm propose that, a pixel belongs to background corresponds to a larger weight with smaller variance. So we can approximate the background by the first B Gaussian distributions which exceed certain threshold T_0:

$$\mathrm{B} = \underset{b}{argmin}\left(\sum_{i=1}^{b} \omega_{i,t} > T_0\right) \quad . \tag{3.2}$$

But in real practice, sometimes if a sequence of the background frames is available, target segmentation will be much easier (in the case of a fixed camera), since we can stop updating the parameters of the Gaussian distribution until a moving object come into the scene, and then just simply compare the coming data with the Gaussian distribution we already built for the background. In Figure 3.1 we will show some Background subtraction result using this MOG model.

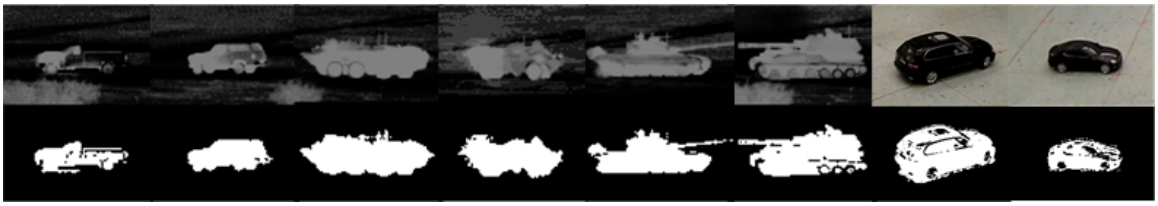


Figure 3.1: Segmentation results, from left to right, the first 6 showing the background subtraction results from SENSIAC database [13], and the last two showing the results from visible data.

3.2 IR Camera Calibration

Since the SENSIAC database which we will be using for our experiments contains a large amount of ground truth data, so here we develop a simple Pinhole model (Figure 3.2) based camera calibration method, in order to do the 3D tracking and recognition. Figure 3.3 will show us the perspective projection of this Pinhole Camera Model.

In Figure 3.3, P' and P are the 2D coordinate and 3D coordinate in the 2D image coordinate system and 3D camera coordinate system respectively. f' is the focal length, and λ is defined as:

$$\lambda = \frac{f'}{z} \quad . \tag{3.3}$$

And then this model can give us the relation between P' and P:

$$\begin{aligned} x' &= \lambda x, \\ y' &= \lambda y, \\ z' &= f' = \lambda z. \end{aligned} \tag{3.4}$$

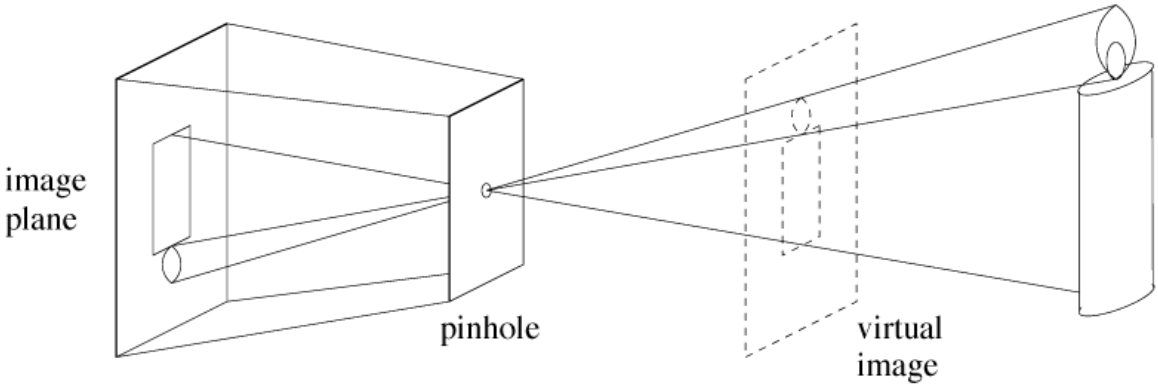


Figure 3.2: Pinhole camera model. *

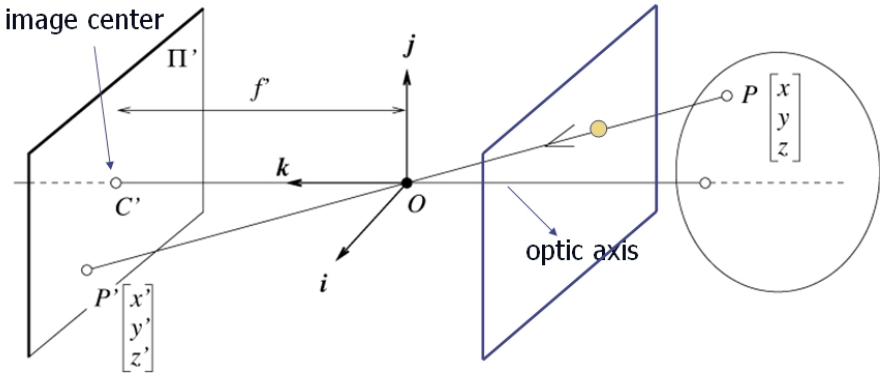


Figure 3.3: Pinhole camera model based perspective projection. *

Since the calibration in SENSIAC dataset which will be used for experiment will be based on the 3D camera coordinate frame, so given the P' at the image center, and the corresponding P in camera coordinate system, we can find the f', then it will be much simple to do the calibration.

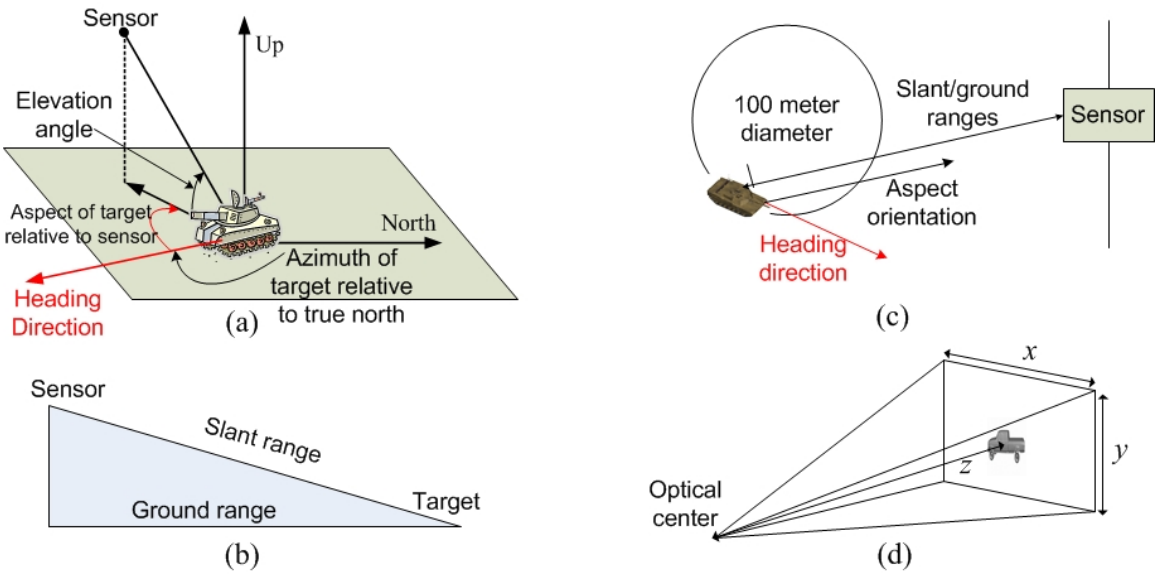


Figure 3.4: Definition of the 3D coordinate system and the spatial geometry of the sensor, and also the target in the SENSIAC dataset. (a) The aspect of the target relative to the sensor, and the azimuth of the target relative to the true north, and the elevation of the sensor; (b) side view of the ground and slant distances between the target and sensor; (c) top-down view of the aspect direction and the heading orientation; (d) A sensor-centered 3D coordinate used for algorithm evaluation.

In Figure 3.4, several ground truth data is displayed in order to calibrate the camera in the camera frame 3D coordinate system, and also for the 3D tracking and error evaluation later. These data includes: the relative position and pointing angle of the sensor to the target, as in Figure 3.4 (a); the ground range and slant range from the target to the sensor, as in Figure 3.4 (b); the pixel location of the center of target in each frame, and aspect and heading direction of the target, as in Figure 3.4 (c); and finally Figure 3.4 (d) the camera frame 3D coordinate system centered at the sensor.

By using the frame which the target's aspect angle is 90 degree to the sensor, we can compute the intrinsic parameter and hence the transformation matrix, and then we will be able to find the 2D pixel location of any given point in the defined 3D coordinate system. The reason of doing this is that we do not have any calibration information from the dataset and neither the true 3D position of the target geometric center.

3.3 Visible Camera Calibration

For the visible video data taken in the lab VCIPL (Visual Computing and Image Processing Lab) at Oklahoma State University, and with the help of the Camera Calibration Toolbox for MATLAB® [48], we can do the camera calibration easily using a check-board pattern, and some markers on the floor.

First we generate and print a check-board pattern, and then paste in on a flat panel. We can download one directly from its website or make our own. In the pattern available online, each square is 30mm×30mm. then we need to generate some calibration images (Figure 3.5) in a folder and then we can get the intrinsic parameters by running the program and extract those corners.

After we get the intrinsic parameters: focal length f, the skew coefficient α_C which defines the angle between the x and y pixel axis, the distortions k_C which store the image distortion coefficients (radial and tangential) as a 5x1 vector, we need to define our 3D world coordinate scene on a flat floor with markers such as Figure 3.6 (a) shows, and then run this program again to obtain the extrinsic parameters. Figure 3.6 (b) shows the 3D coordinate system after calibration, and the re-projected points.

In order to verify the calibration, we can simply define a 3D cube in the world coordinate system, and then use the calibration information we got from the previous procedures introduced to

compute the 2D pixel location, and then show this cube in the image of the 3D coordinate system, as shown in Figure 3.7.

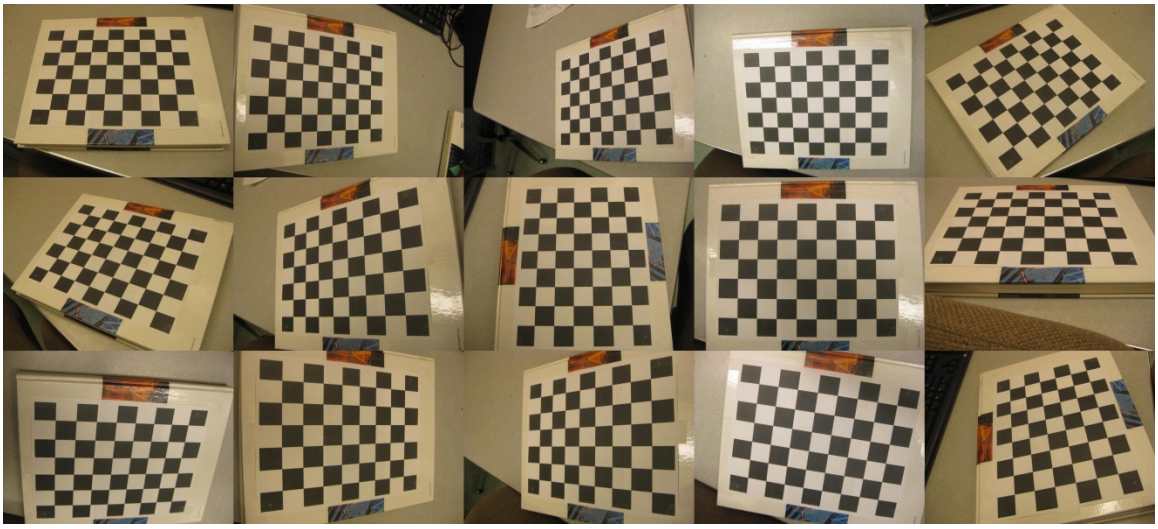
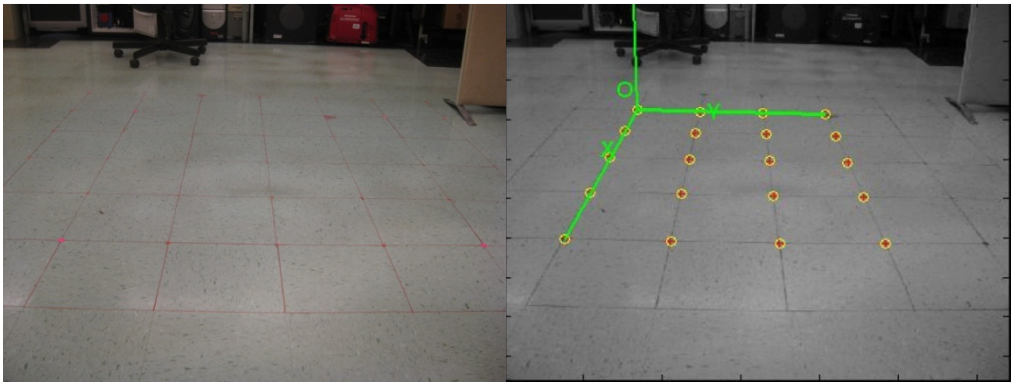


Figure 3.5: Image of pattern board for intrinsic camera calibration.



(a) (b)

Fig.3.6. Extrinsic camera calibration (a): Real 3D scene of a flat floor plane. (b): 3D coordinate system after camera calibration.

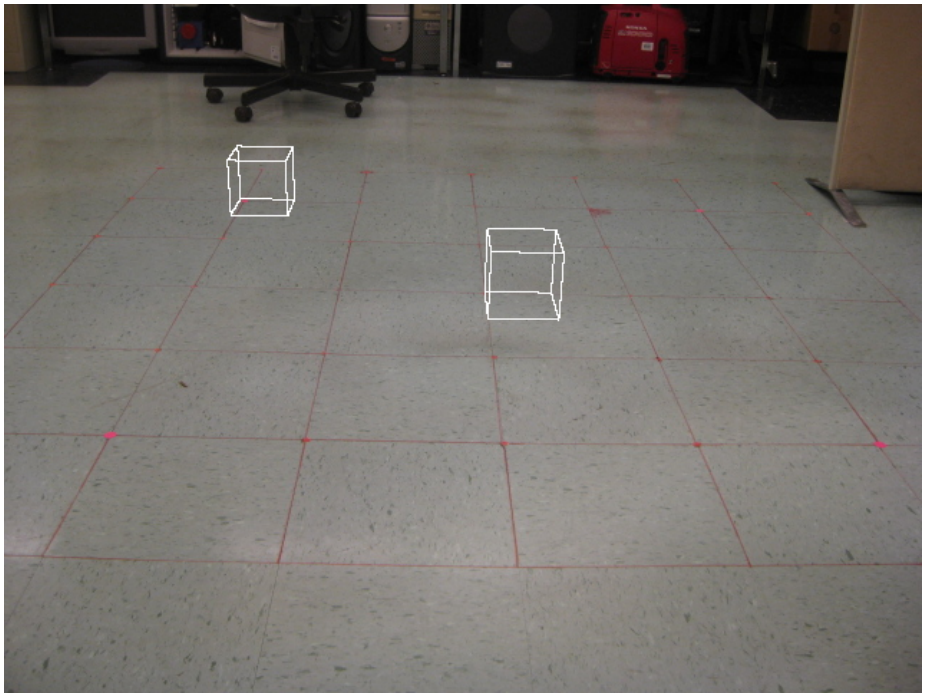


Figure 3.7: Two white cubes (140x140 mm) drawn in the 3D world coordinate system using camera calibration data computed earlier; the further-lower-left corner of the left cube lies at the origin, and the further-lower-left corner of the right cube lies at (610,610,0) in the world coordinate system in millimeter. Note that the dimension of the bricks in the floor is305x305 in mm.

---

* Figures taken from http://www.cs.gmu.edu/~zduric/it835/Slides/Cameras.pdf

CHAPTER IV

SHAPE BASED GENERATIVE MODEL

The key of the generative appearance model will be shown in Figure 4.1. The first step of the model learning is to obtain group of silhouettes from a set of targets which belong to several different classes that are observed from different viewpoints. Then, we will try to learn a mapping from the high dimensional silhouettes space to tow low dimensional manifold which contains only the information of the variations in terms of pose and identity. After this model is learned, we can reconstruct or interpolate a semantically meaningful silhouettes image given any arbitrary point on each of the LD manifolds. In this chapter, we will introduce identity and view manifolds, and then the non-linear tensor decomposition method which can combine the identity and view manifold together into a generative model for multi-view target shape modeling, and finally the target tracking and recognition. This work was previously proposed by Dr. Vijay Venkataraman [4]. But in this paper we expand it to include more training targets, not only civilian vehicles but also military vehicles, and we will expand the application of this generative model to the IR database ATR purpose and some other visible band data.

4.1 Preliminary

The generative model uses two very important mathematical tools, which I would like to introduce first in this section.

4.1.1 RBF Mapping

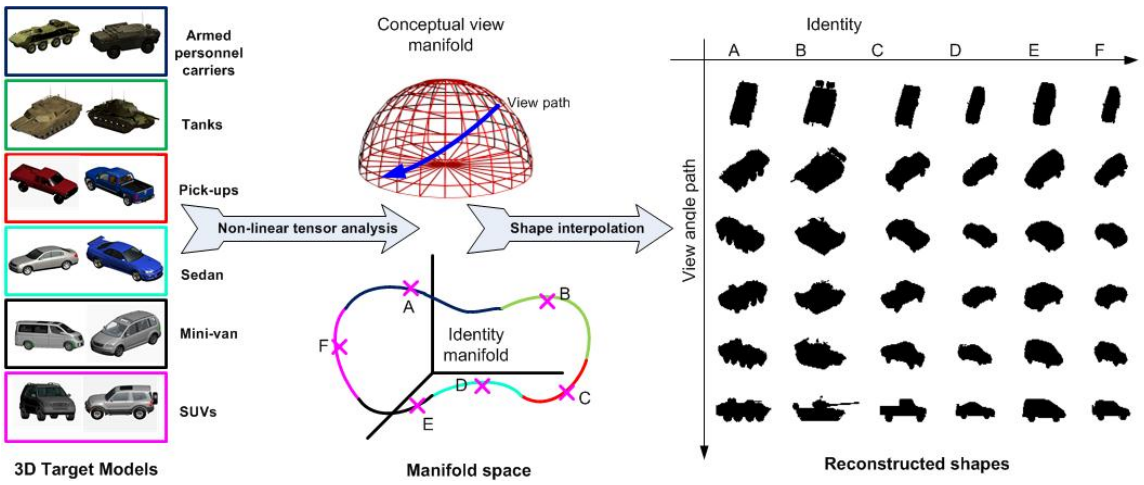


Figure 4.1: Illustration of the generative model for view and identity based shape appearance synthesis. Reconstruction results of the shape are shown for the blue path traversed along the view manifold and for six different points on the identity manifold. In each case the reconstructed shape has strong characteristics of the view and target class.

The first tool which will be used is called the radial basis functions (RBF), this is a non-linear kernel function that is normally used for function approximation, interpolation, etc. for example as shown in Figure 4.2, just knowing the function values at the black markers, we can approximate the function by placing radial basis kernel at the known values and interpolate to other unknown regions. The advantage that RBF can offer is a smooth function approximation that preserves local similarity.

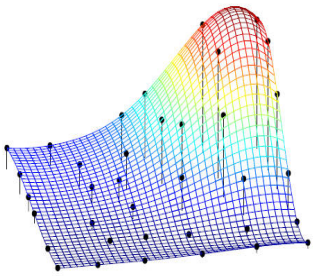


Figure 4.2: RBF mapping and interpolation *.

4.1.2 Tensor

Another tool we will use is called tensor decomposition or higher order singular value decomposition (HOSVD) [38]. It is similar in spirit to the singular value decomposition or SVD where we identify the basis vectors of a given matrix. HOSVD allows us to identify the basis vector of a higher dimensional data vector. For example as shown in Figure 4.3, we have face images arranged in 3D, with each dimension representing one of the following (identity, view, illumination). HOSVD allows us to identify the basis vectors of each of these subspaces or factors. But this is a linear decomposition.

Although the application of common linear algebra techniques like principal component analysis (PCA) and independent component analysis (ICA) have been successfully used in lots of image analysis and problems mainly in the field of face recognition [49], most of these methods are built for the analysis of a single varying factor. In research of face recognition usually the factor means the individual identity. Therefore these methods have a lot of issues in decomposing other varying factors like pose or illumination when they are encountered in the dataset. So, we are trying to develop a target appearance model by which the identity and view information can be decomposed into different factors, this method is combined with the non –linear RBF kernel, resulting in a non-linear tensor decomposition method [4]. Now we will introduce the tensor as followed, which was previously talked about in Dr. Vijay's PHD dissertation *'Advanced Machine Learning Approaches for Target Detection, Tracking and Recognition'*, 2010.

A **tensor** is defined as a multi-dimensional matrix or a n-way array or n-mode matrix. They are higher order generalizations of a vector (first order tensor) and a matrix (second order tensor). Here we will indicate scalars by lower case letters ($a, b, \ldots$), vectors by bold lower case letters ($\boldsymbol{a}, \boldsymbol{b}, \ldots$), matrices using bold upper case letters ($\boldsymbol{A}, \boldsymbol{B}, \ldots$), and higher order tensors by script upper case letters ($\mathcal{A}$, $\mathcal{B}$,...), the order of a tensor $\mathcal{A} \in R^{I_1 \times I_2 \times \ldots \times I_N}$ is N and an element of this

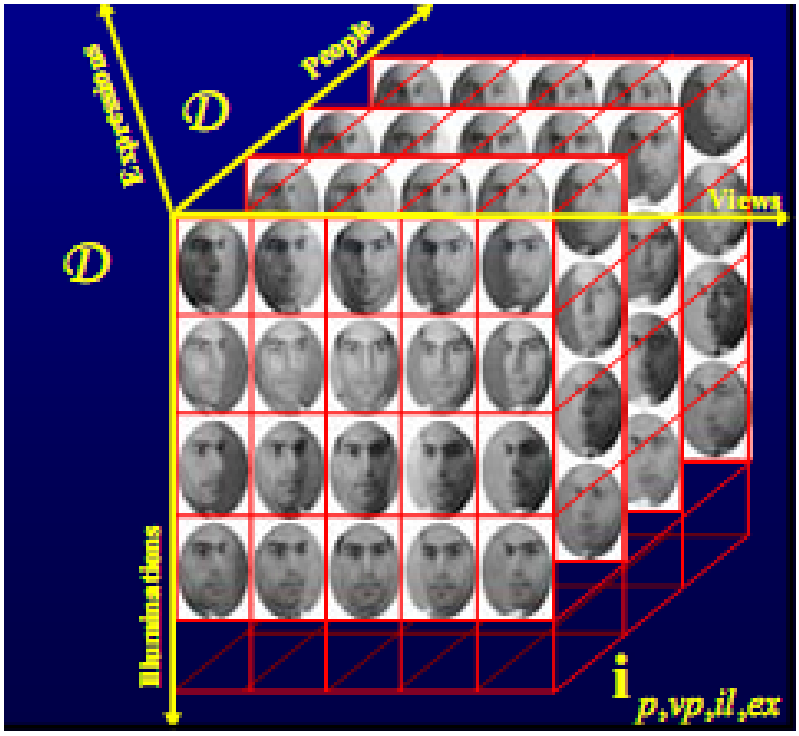


Figure 4.3: Example of HOSVD [38].

tensor is denoted by $a_{i_1 i_2 \ldots i_N}$, where $1 \le i_n \le i_N$.

Tensor Flattening: Flattening is one of the most significant operation that will be performed on tensor, or we can call it matricization or unfolding, it is the reordering of the elements of a tensor into a 2D matrix. For example, a $2 \times 3 \times 4$ tensor can be re-arranged into a 4×6 matrix, 3×8 or a 2×12 matrix. The mode-n matricization of a tensor $A \in R^{I_1 \times I_2 \times \ldots \times I_N}$ is denoted by $\boldsymbol{A}_{(n)}$. The example provided in [50] is repeated here for clarity of the flattening concept. Let the frontal slices of $A \in R^{3 \times 4 \times 2}$ can be represented by

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \boldsymbol{A}_2 = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix}, \tag{4.1}$$

then the three mode-n flattened matrices are given by

$$\boldsymbol{A}_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix}, \tag{4.2}$$

$$\boldsymbol{A}_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 23 \end{bmatrix}, \tag{4.3}$$

$$\boldsymbol{A}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & \dots & 11 & 12 \\ 13 & 14 & 15 & \dots & 23 & 24 \end{bmatrix}. \tag{4.4}$$

Keep in mind that different paper or work use various ordering of the columns for the n-mode flattening. But in general, the particular matrix permutation of the columns is not important only if it is compliant along all related calculations. [50]

Mode-n product: the next thing involving tensors that is of our interest is the mode-n product. The mode-n product of a tensor $A \in R^{I_1 \times I_2 \times \dots \times I_n \times \dots \times I_N}$ with a matrix $Q \in R^{J \times I_N}$ is denoted by $B = A \times_n Q$. The tensor $B \in R^{I_1 \times I_2 \times \dots \times I_{n-1} \times J \times I_{n+1} \dots \times I_N}$ and element-wise we have

$$b_{i_1 \dots i_{n-1} j i_{n+1} \dots i_N} = \sum_{i_n=1}^{I_n} a_{i_1 i_2 \dots i_n} q_{j i_n}, \tag{4.5}$$

The mode-n multiplication may be represented using flattened matrices as follows

$$B = A \times_n Q \Leftrightarrow B_{(n)} = Q\boldsymbol{A}_{(n)}, \tag{4.6}$$

Another important property of the mode-n product is

$$A \times_n P \times_m Q = A \times_m Q \times_n P \ (m \neq n). \tag{4.7}$$

We will discuss about the tensor decomposition later in this chapter.

4.2 Identity Manifold

Figure 4.4: LD identity manifold

Not like many other methods in terms of the way how they handle the identity manifold, it is possible for us to interpolate and reconstruct any unknown subclass vehicle information given a limited number of training data. The continuous nature of the identity manifold will give us great help to interpolate meaningful target silhouetted between two known training targets. In this section we are going to introduce an identity manifold which plays an important role in our target tracking and recognition algorithm where both inter-class and intra-class shape are captured.

However, since we want a LD latent space which contains only the identity factor (Figure 4.4) rather than the HD space in which both view and identity information are coupled together with each other, and moreover, we are trying to build a semantically valid and meaningful identity manifold that can support us to do the interpolation which significantly make sense in the real world between class and within class, so it will be very important for us to answer two questions: the first one is where and which should we learn this manifold and the second one is, how do we

learn this semantically meaningful manifold or what kind of property should we keep on the designated manifold.

We will discuss about the first question in section 4.4. Here we will focus on the second one, which is related to how we determine the topology of the identity manifold, and also the ordering or topological relationship of those training data. If the training data are sparse, a 1D closed-loop structure was suggested to ensure valid interpolation on identity manifold and in this way we can ease the inference process later for the vehicle tracking and recognition. It starts with a few considerations which seems arbitrary but actually due to some practical issues. First, the learning of a large number of the dataset of targets may not be necessary for a specific ATR application. And this is also the motivation for us to interpolate the vehicle shape information between and within class. The second one is, this identity manifold is assumed to be open because all the training vehicles are artificial and share some common shape information to each other, instead of extremely different. And the third one is that the closed-loop structure will facilitate the inference process, preventing the hypothesis from going to a dead end on the structure.

Since now we have determined the topology of the identity manifold, the next step is how do we order the data on this manifold. Although there is no well-defined ordering relationship among different vehicles, we want vehicles within the same class stay close to each other, so here a class-constrained shortest-closed-path method was suggested in order to find this specific ordering across all training vehicles along the identity manifold. But this method requires view independent dissimilarity between identities which should be obtained from 3D models, but for convenience, to compute the dissimilarity we talked about, we simply use the accumulated mean square errors of multi-view silhouettes after the distance transform.

In detail, assume we have a training set of target shapes from N view independent identity vectors in a LD space i^k, and $k \in \{1,...,N\}$, and L_k is the associated class labels. y_m^k is the

silhouette of object k with view m after vectorization and distance transform [33]. Here we denote target u and v by i^u and i^v, then the similarity between these two targets over M number of training views is defined as:

$$D\left(i^u, i^v\right) = \sum_{m=1}^{M} \left\| y_m^u - y_m^v \right\| + \alpha \cdot \varepsilon\left(L_u, L_v\right) \quad , \tag{4.8}$$

where the

$$\begin{aligned} \varepsilon\left(L_u, L_v\right) &= 0 \quad if\ L_u = L_v, \\ &= 1 \quad otherwise, \end{aligned} \tag{4.9}$$

and the $\|\cdot\|$ here denotes the Euclidean distance, and α is a constant. The term $\varepsilon\left(L_u, L_v\right)$ is a penalty term makes sure that targets within the same class will stay together next to each other on the identity manifold. Since targets in the same class will look closer to each other, and this can give us some reasonable interpolation within class. Otherwise if we put all training targets in arbitrary orders, most probably we will get a lot of interpolated targets that do not exist in the real world.

Next we denote the manifold topology by

$$T = \left[t_1, t_2, ..., t_{N+1}\right] \quad , \tag{4.10}$$

where $t_i \in \left(1, N\right]$, $t_i \neq t_j$, for $i \neq j$ and $t_1 = t_{N+1}$. Then we can write down the class-constrained shortest-closest-path as

$$T = \underset{T}{argmin} \sum_{i=1}^{N} \left(i^{t_i}, i^{t_{i+1}}\right) . \tag{4.11}$$

This topology tends to group all those objects of the similar 3D shapes together, since the penalty term makes sure targets within same class stay next to each other, so we can guarantee the best local smoothness along this identity manifold, and this is very essential for valid identity interpolation of some unknown sub-class.

4.3 Conceptual View Manifold

We need to design a view manifold which yields the view-leaded shape variability to each target except from the identity manifold, or independent from the identity manifold. Some common ways were found using non-linear dimension reduction algorithms such as LLE or Laplacian Eigenmaps, trying to find the LD view manifold given the high dimensional observations for a particular target [33]. But two main defects exist here. One is, since they are identity dependent, so the involved multiple view manifolds will be aligned together for general multi-view target modeling in the same latent space. The other one is, they are usually constrained by the 1D structure which is too simple to capture all possible object poses in the real world. So, we will develop a hemisphere-like view manifold which will include almost all those possible view angles around a vehicle as shown in Figure 4.5. This view manifold is characterized by two parameters: the azimuth/ aspect angle θ, and the elevation angle φ. This conceptual view manifold can help us to stay away from the learning and also aligning of multiple view manifolds for various targets. And simultaneously, it will also provide us a unified and intuitive viewing space representation, and give us strong supports on efficient dynamic view estimation since this conceptual view manifold is continuous and it contains most of the possible viewports we will meet during training and testing.

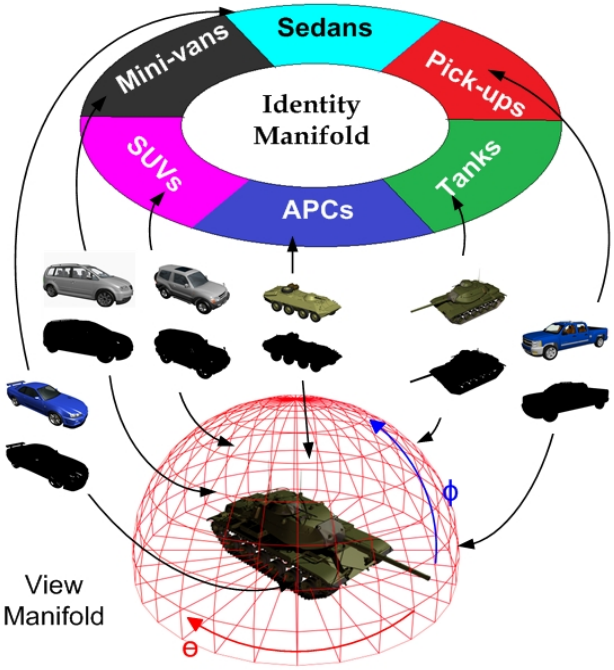


Figure 4.5: Coupled view-identity manifolds for shape based multi-view target modeling. Two factors, identity and view can be decomposed from the shape variations in the training dataset, and both of them can be mapped to a low dimension (LD) manifold. So by selecting a point on each of the manifold, we can interpolate a new shape through the reconstruction which we will talk about later.

4.4 Non-linear Tensor Decomposition

We will extend the non-linear tensor decomposition method that was described in [34] to develop the shape-based generative target model, which is controlled by only three variables. This will involve learning a non -linear mapping function that will find the relationship from the HD observation to the unified view manifold and then by given a view independent space, factorizing out the identity vector for identity representation (the first issue we mentioned in 4.1).

For the target number k, we will learn a non –linear mapping between each of them by the generalized radial basis function (GRBF) kernel as:

$$y_m^k = \sum_{l=1}^{N_C} w_l^k \kappa\left(\left\|\Theta_m - \Upsilon_l\right\|\right) + \left[1\ \Theta_m\right] b_l \ , \tag{4.12}$$

here we denote the d-dimensional observation of target k under view m by $y_m^k \in \mathcal{R}^d$, and $\Theta_m = \left[\theta_m, \phi_m\right]$ is the LD point of view m on the view manifold, where $0 \le \theta_m \le 2\pi, 0 \le \phi_m \le \pi/2$. $\left\{\Upsilon_l \mid l = 1, ..., N_C\right\}$ are N_C kernel centers on the view manifold, w_l^k is the specific target weights of each kernel and b_l is the mapping coefficient of the linear polynomial $\left[1\ \Theta_m\right]$ included for regularization. $\kappa(\cdot)$ is the common Gaussian Kernel.

This mapping can be written in a matrix form as:

$$y_m^k = B^k \Psi\left(\Theta_m\right) \ , \tag{4.13}$$

B^k is a $d \times (N_C + 3)$ linear mapping corresponding to target k and containing the weight terms w_l^k in Eq.4.12, and where $\Psi\left(\Theta_m\right) = \left[\kappa\left(\left\|\Theta_m - \Upsilon_1\right\|\right), ..., \kappa\left(\left\|\Theta_m - \Upsilon_{N_C}\right\|\right), 1, \Theta_m\right]$ is a non-linear kernel mapping that composed of the regularization term $\left[1\ \Theta_m\right]$. Since $\Psi\left(\Theta_m\right)$ is dependent only on the view angle, it will be easier for us to understand that, the identity related information is included in B^k . For K number of training vehicles, we might want to obtain their corresponding mapping B^k for each $k = \{1, 2, ..., K\}$ that could be stacked together later to form a 3D tensor $\mathrm{C} = \left[B^1 B^2 ... B^K\right]$, which contains all those identity –dependent information belong to different view angles, or poses. Intuited by the fact that singular value decomposition is usually used to find the basis vectors of its row and column spaces, the application of the high-order singular value decomposition (HOSVD) [51] to this tensor C could help us to factorize out K

identity vectors $i^k \in \mathcal{R}^K$, which can be interpreted as the basis vectors of a latent space for the identity factor. And this will give us the following tensor decomposition:

$$\mathrm{C} = \mathrm{A} \times_3 i^k \quad , \tag{4.14}$$

where $k = \{1, 2, ..., K\}$, A is a $d \times (N_C + 3) \times K$ core tensor that serves as coupling the identity and view factors together and $\times_j$ denotes the mode- j tensor product.

Then from Eq.4.13 we will be possible to reconstruct the training target silhouettes corresponding to the k^{th} target under view m:

$$y_m^k = \mathrm{C} \times_2 \Psi(\Theta_m) \; . \tag{4.15}$$

It is also possible for us to interpolate the shape silhouette given any Θ on the view manifold and an interpolated i between training identity vector i^k. Because of the RBF kernels' nature of possibly interpolation nature, we can do this reconstruction between training target and training views. And furthermore, since this kind of interpolation can be attribute to the sparse nature of the training dataset in terms of the identity variation, so it is hard to say that given any arbitrary identity vector i which span the basis vectors $(i^1, ..., i^k)$ in the tensor coefficient space.

Another point is, one may think any linear combination of these basis vectors could form a new identity vector, and this vector may lead to a meaningful shape interpolation. But the shape produced in this way usually does not represent a real world object. A LD space that is appropriately supported by all training targets is needed to ensure valid shape reconstruction. This motivates us to learn a 1D closed-loop identity manifold (Figure 4.6) via B-spline curve fitting (Figure 4.7) in the tensor coefficient space $\{i^k \mid k = 1, ..., K\}$ according to the manifold topology defined in Eq.4.11. We constrain this identity space to include only those points that lie on this

closed B-spline curve connecting basis vectors according to the manifold topology. This closed loop 1D spline curve is defined as the identity manifold. It is expected that an arbitrary identity vector along this identity manifold would be more semantically meaningful due to its proximity to training targets, and it should support valid shape interpolation on the manifold. Thus Eq.4.14 is controlled by only two continuous variables in a LD latent space, and each of them follows its own manifold, to form a multi-view shape model which will constrain the smoothness of the

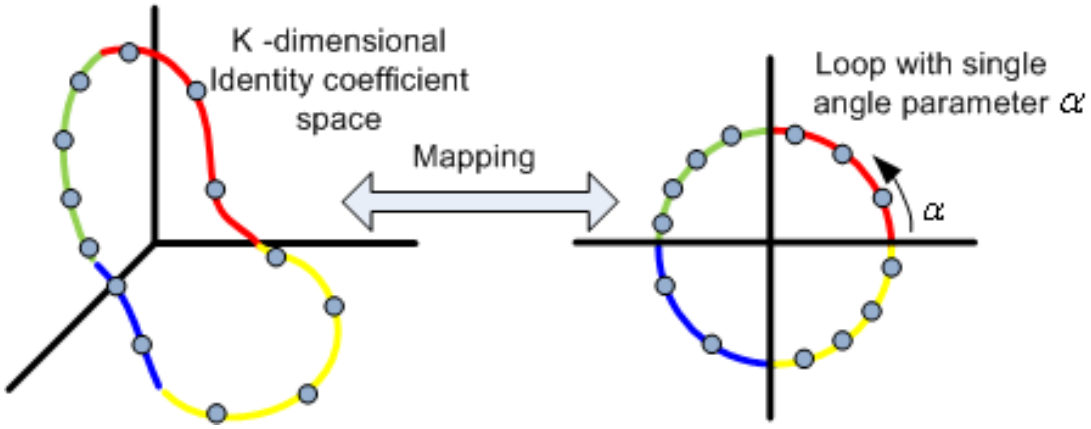


Figure 4.6: Mapping from the K-dimensional identity coefficient space to the 1D closed loop controlled by one single angle parameter α .

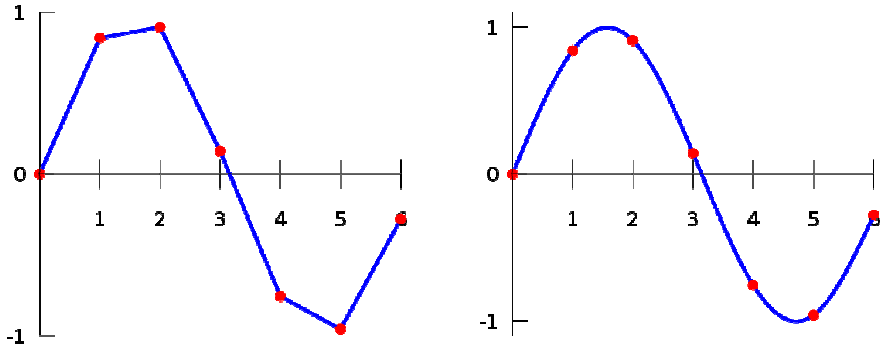


Figure 4.7: Left: linear interpolation. Right: Spline interpolation. **

interpolation transition between different targets. Due to the reason that we constrain the space of the identity to be a closed spline in $\mathcal{R}^K$, we can always find a one to one mapping from points

on this spline to those points on a unit circle controlled by a single angular variable $\alpha \in [0, 2\pi)$. This will significantly simplify the inference process on the identity manifold, by using only one single variable instead of the identity vector. Now, the Eq.4.14 can be generalized to reconstruct valid shape silhouettes corresponding to arbitrary points along both the view and identity manifolds:

$$\mathrm{C}_{\alpha} = \mathrm{A} \times_3 i(\alpha) \ , \tag{4.16}$$

and then Eq.4.15 can re-written as:

$$y(\alpha, \Theta) = \mathrm{C}_{\alpha} \times_2 \Psi(\Theta) \ . \tag{4.17}$$

Thus we can generalize the above two equations as:

$$y(\alpha, \Theta) = \mathrm{A} \times_3 i(\alpha) \times_2 \Psi(\Theta), \tag{4.18}$$

where $i(\alpha)$ is a specific identity vector along the identity manifold. So, Eq.4.17 defines a new target generative model for multi-view shape modeling, which is only controlled by two continuous variables, α and Θ, each of which are defined along their own manifold. Due to the fact that the identity manifold is a closed loop spanning in the ND coefficient space, we can map it to a circle in order to make the inference easier.

In the following chapter, we will begin to talk about our inference algorithm and how we utilize our generative model and integrate both of the view and identity manifold in to real applications.

---

* Figure taken from http://www.algebra-cheat.com/radial-basis-functions-for-simulating-pdes.html

** Figures taken from http://en.wikipedia.org/wiki/Interpolation

CHAPTER V

JOINT TRACKING AND RECOGNITION

In this chapter we will go through the most important part for our application of this generative model we introduced in earlier chapter, into the ATR purpose. We first introduce the motion model we used during the tracking and recognition process, and then we will talk about the obtaining of target silhouettes for training, and hence the particle filter-based inference algorithm.

5.1 Maneuvering Target Motion Models

The white noise acceleration model is probably the simplest model for target maneuvering with two -dimensional per coordinate. And this is a widely used model derived from simple equations of motion: constant acceleration and constant velocity [52]. It is general in the explanation that the second and third order derivatives of the position are not zero, but a zero-mean random process. In our application, since all vehicles are driving on the round and there is no big change of acceleration, we decide to use this simple white noise motion model defined below:

$$\begin{aligned} \theta_t &= \theta_{t-1} + w_t^{\theta}, \\ v_t &= v_{t-1} + w_t^{v}, \\ x_t &= x_{t-1} + v_{t-1}\sin(\theta_{t-1})\Delta t + w_t^{x}, \\ y_t &= y_{t-1} + w_t^{y}, \\ z_t &= z_{t-1} + v_{t-1}\cos(\theta_{t-1})\Delta t + w_t^{z}, \end{aligned} \tag{5.1}$$

where Δt is the time interval between two adjacent frames. This motion model assumes that the process noise coupled with the target's kinematics follow a Gaussian distribution, so we define

that $w_t^{\theta} \sim N(0,\sigma_{\theta}^2)$, $w_t^{v} \sim N(0,\sigma_{v}^2)$, $w_t^{x} \sim N(0,\sigma_{x}^2)$, $w_t^{y} \sim N(0,\sigma_{y}^2)$, $w_t^{z} \sim N(0,\sigma_{z}^2)$, and these variances can be chosen flexible to accommodate various target dynamics and ground conditions. Which means, for example, if a target is moving fast and making turn fast, a larger variance could be used, and for the case if the ground condition is uneven and rough particularly a larger σ_y^2 will be needed. This motion model will be clearly shown in Figure 5.1.

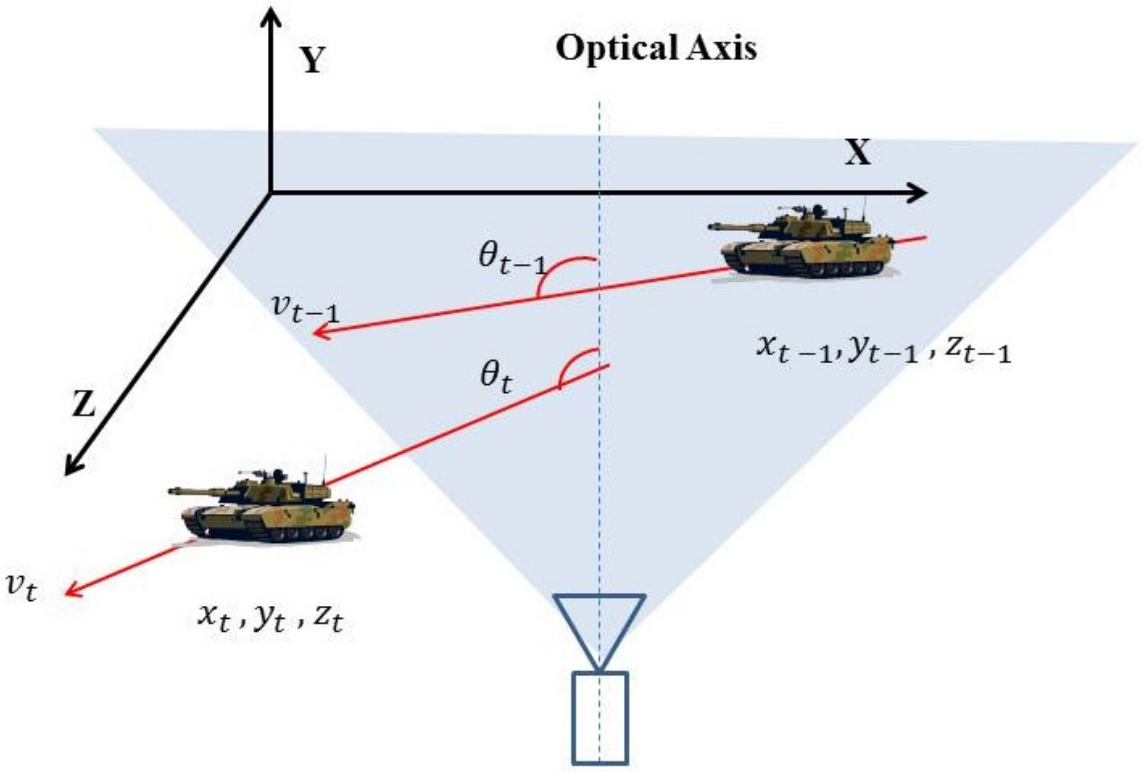


Figure 5.1: Motion model.

5.2 Target Silhouettes Used for Training

Before we train our generative model, one important thing is to obtain all those silhouettes at different training viewpoints from vehicles we are interested. But since it is inaccurate and impossible for us to do this in the real world, we will use a camera in a virtual world to create all the training silhouettes we need, in order to train the generative model.

The easiest way is to set up a camera and load a fixed 3D model in 3Ds Max and then render all silhouettes at a sequence of training camera positions, which is equivalent to the way when camera is fixed and the pose of target is changing. But due to the reason that we want to save time and build all these training silhouettes efficiently, we will use XNA in Visual Studio C# to render them. It is the same idea but we can choose to change the pose of the target at all training viewpoints to build the silhouettes. Figure 5.2 will show how this is done in XNA. But due to the fact that the distance from target to sensor/camera is relatively large, we will use orthographic projection in practices.

Figure 5.2: Obtaining training silhouettes from 3D models in XNA.

5.3 Inference algorithm

A generative graphical model is used to combine all the factors together with their conditional dependencies into a framework which is probabilistic, and hence to estimate the state of the target including 3D location and category sequentially from a dataset of target silhouette $\{\mathbf{Z}_t \mid t = 1,...,T\}$, which could be obtained from background subtraction that we discussed in earlier chapters. We show this probabilistic graphical model in Figure 5.3. $\mathbf{X}_t = [x_t\, y_t\, z_t\, \theta_t\, v_t]$ is the state vector containing the targets 3D position along the horizontal, vertical and range orientations, the heading direction with respect to the optical axis and the speed in a 3D coordinate system as shown in Figure 5.4. P_t is the camera projection matrix, and since in all of

our dataset, the camera is static, so we could set that $\mathrm{P}_t = \mathrm{P}$. Another variable α_t indicates the identity position on the identity manifold.

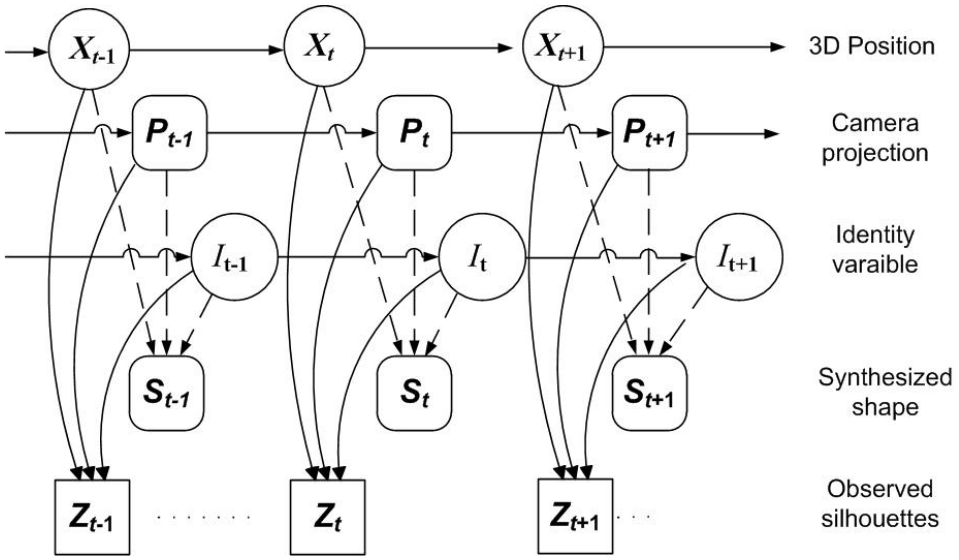


Figure 5.3: ATR inference graphical model.

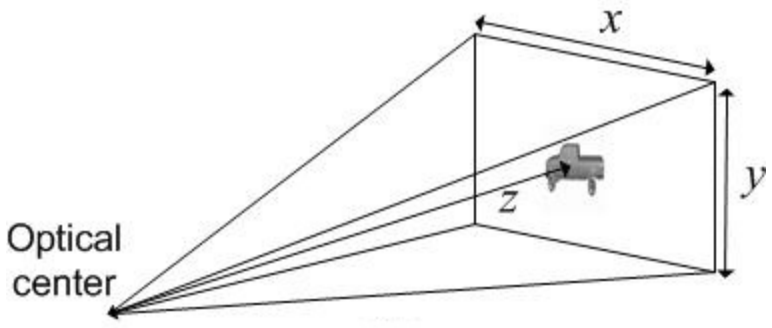


Figure 5.4: Sensor centered 3D coordinate system.

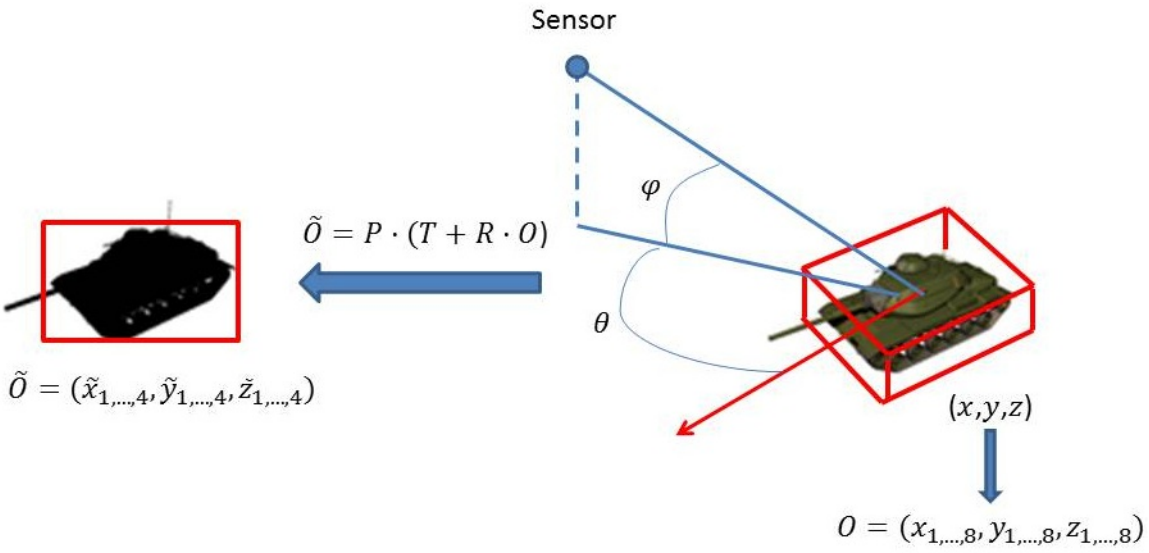


Figure 5.5: Illustration of the 3D-2D projection

Moreover, the generative model defined in Eq.4.14 needs another parameter Θ to synthesize the target shape $\mathcal{Y}_t$ that can be computed from $\mathbf{X}_t$ and P_t. Since our target silhouettes are obtained from a fixed range virtual camera, we need to scale all of those $\mathcal{Y}_t$ appropriately in order to accommodate all different camera distances in the real data sequences. To summary, the synthesized $\mathcal{Y}_t$ is a function of those three factors: α_t, P_t and $\mathbf{X}_t$.

Specifically, as shown in Figure 5.5, given the location of the target centroid center (x, y, z) and the dimension of the vehicle, we can always find the corresponding 3D locations of the 8 corners of a bounding cube around the target: $O = (x_{1,\dots,8}, y_{1,\dots,8}, z_{1,\dots,8})$, then from O we can find the translation matrix T of the cube from the sensor center. And, from the aspect angle θ and the target's azimuth angle relative to the north, and also the target 3D location, we can find out the rotation matrix R. So together, we can find out the 2D image pixel location of a bounding box around the target in the image: $\tilde{O} = P(T + RO)$, where $\tilde{O} = (\tilde{x}_{1\dots4}, \tilde{y}_{1\dots4}, \tilde{z}_{1\dots4})$ Note that we will only need four corners' 2D image location in order to determine this 2D bounding box.

Given an observation, the target silhouette $\mathbf{z}_t$ from the background subtraction result of the data, the automatic target tracking and recognition problem becomes the sequential estimation of the posterior probability of $p(\alpha_t \mid \mathbf{X}_t, \mathbf{z}_t)$. Because of the non-linear nature of this inference problem, the particle filter approach [53] is applied here, and it needs the dynamics of the two variables $p(\mathbf{X}_t \mid \mathbf{X}_{t-1})$, and $p(\alpha_t \mid \alpha_{t-1})$, and also the likelihood function $p(\mathbf{z}_t \mid \alpha_t, \mathrm{X}_t)$. The condition for P_t is omitted because we assume that the camera is static.

Given the hypothesis of $\mathbf{X}_t$ and α_t in the i^{th} frame as well as the P_t, we can find the corresponding shape information (silhouettes) $\mathbf{y}_t$ from the generative model Eq.4.17 with additional scaling factor depends on the actual range of the target from the sensor. Then the likelihood function which measures the similarity between the hypothesis $\mathbf{y}_t$ and the observation $\mathbf{z}_t$ will be defined as:

$$p(\mathbf{z}_t \mid \alpha_t, \mathbf{X}_t) \propto \exp[-\frac{\left\|\mathbf{z}_t - \mathbf{y}_t\right\|^2}{2\sigma^2}], \tag{5.2}$$

where the σ^2 controls the sensitivity and $\| \ \|^2$ gives the mean square error between the hypothesis and the observation shape.

During the joint tracking and recognition, although the target's identity does not change like the pose, the estimation of identity along the identity manifold should be varying due to the uncertainty and ambiguity of the observations. Hence we define our dynamic along the identity manifold as a simple random walk:

$$\alpha_t = \alpha_{t-1} + w_t^{\alpha} \ , \tag{5.3}$$

where $w_t^{\alpha} \sim N(0, \sigma_{\alpha}^2)$. This simple model will give the estimation of identity some freedom to evolve along the identity manifold and hence converge to the correct class during the sequential estimation.

Table 5.1: Pseudo-code of the particle filtering-based ATR algorithm

---

- Initialization: Draw $\mathbf{X}_0^j \sim N(\mathbf{X}_0, 1)$, and $\alpha_0^j = \alpha_0$, $\forall j \in [1, ..., N_P]$. Here $\mathbf{X}_0$ and α_0 are the initial kinematic state and identity values respectively.
- For $t = 1, ..., T$ (number of frames)
  1. For $j = 1, ..., N_P$ (number of particles)
     1.1 Draw $\mathbf{X}_t^j \sim p(\mathbf{X}_t^j \mid \mathbf{X}_{t-1}^j)$ samples
     and $\alpha_t^j \sim p(\alpha_t^j \mid \alpha_{t-1}^j)$ as in Eq.5.1 and Eq.5.3.
     1.2 Compute weights $w_t^j = p(\mathbf{z}_t \mid \alpha_t^j, \mathbf{X}_t^j)$ using Eq.5.2.
     End
  2. Normalize the weights such that $\sum_{j=1}^{N_P} w_t^j = 1$.
  3. Compute the mean estimation of the kinematics and identity using $\hat{\mathbf{X}}_t = \sum_{j=1}^{N_P} w_t^j \mathbf{X}_t^j$, and $\hat{\alpha}_t = \sum_{j=1}^{N_P} w_t^j \alpha_t^j$.
  4. Set $[\alpha_t^j, \mathbf{X}_t^j] = \text{resample}(\alpha_t^j, \mathbf{X}_t^j, w_t^j)$
- End

CHAPTER VI

EXPERIMENTAL RESULTS

In this chapter we will discuss in details about the experiments (ATR) we did in order to show the practical application of the generative model we introduced in previous chapters. Specifically, we have developed four different kinds of ATR algorithms which share the same particle filtering inference framework shown in Figure 5.3. Through these four different methods, we will evaluate the effectiveness of shape silhouettes interpolation on both view and identity manifold.

Method I uses the recently proposed generative model we introduced that involves both the view and identity manifold, and we will use the interpolated silhouettes on both two manifolds as the target appearance model, which means both the view and identity manifolds are continuous. Method II and III are simplified versions of the appearance model in which the silhouettes information is interpolated only on the view, and identity manifold respectively (i.e., in method II, the view manifold variable is continuous and the identity manifold variable is discrete, while in method III, the identity variable is continuous and the view manifold variable is discrete). In method IV, we will only use the training silhouettes as appearance for shape matching without any shape interpolation. This means both the view and identity manifold variables are discrete. These four different methods will basically show us the effectiveness of the interpolation on view, and identity manifold throughout the real ATR application.

Three major experiment results will be shown next. These involve the tracking and recognition

based on IR video data from SENSIAC dataset, and also on the visible data. Firstly we will show the learning of the recently proposed generative model, demonstrated by the simulation results of shape interpolation on view and identity manifold respectively. Then we introduce the SENSIAC dataset [13] followed by experimental results on a set of IR video sequences of eight different targets at six different ranges. Then, we also included three different visible video data sequences for our algorithm evaluation. These visible data contains two sequences that were captured in VCIPL at Oklahoma State University from two remote-controlled toy vehicles, and another one from a real-world surveillance video provided from PETS2001 dataset [14]. Before the ATR algorithm, some pre-processing (Background subtraction [47] as shown in Figure 6.1) was applied to the data in order to get the initial background subtraction result in each frame, and also the distance transform [33] was used to generate the observation data that were once used for the learning of the generative model.

6.1 Learning of the Generative Model

First we collected thirty-six 3D models that can be handled in 3Ds Max, these models contains 6 models for each of the 6 target types: Tanks, APCs (Armored Personnel Carriers), Pick-ups, mini-vans, cars and SUVs for the model learning. All these models are shown in Figure 6.2. All these 3D models are scaled to the same size as they are in the real world, except those tanks and APCs, which are 0.8 times the real size in the real world, due to the fact that vehicles in these two classes are much more larger than the other four classes. Such kind of size modulation and category dependent scaling can help us to learn the unified generative model and then to estimate the real distance information in a scene. For each 3D model, we generated a set of silhouettes with respect to training viewpoints previously chosen on the view manifold, as discussed in 5.2. For convenience, we only consider the azimuth angle that ranges $0 \le \theta < 2\pi$, and the elevation angle ranges $0 \le \varphi < \pi/4$.

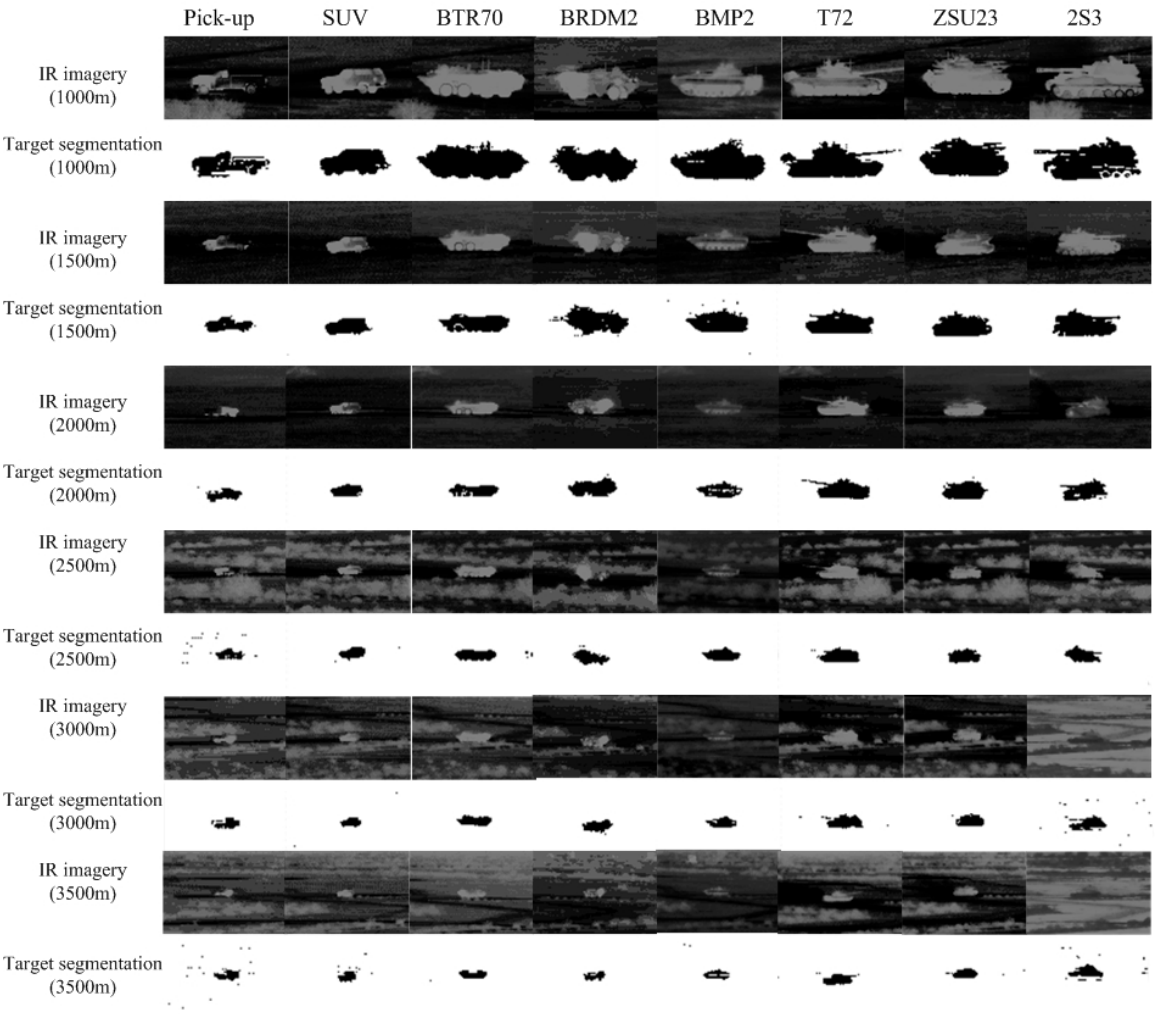


Figure 6.1: Group of snapshots of the segmentation results for all eight targets at all six different ranges together with the original IR sequences.

Because of the computational load, and the fact that most of the view in our experimental data are below elevation angle of $\pi/4$, and also the reason that shape silhouettes variety changes smaller and smaller as the elevation angle φ goes larger and larger to the top of the hemisphere view manifold. This will lead to a non-uniformly sampled view manifold. Specifically, 150 training

Figure 6.2: All 3D models that have been used for model learning. Each target class contains six. From left to right: APCs, Tanks, pick-ups, cars, mini-vans, SUVs, in the order according the identity manifold from top-down and left to right.

Figure 6.3: Shape interpolation on the view manifold, where the left, middle and the right ones indicates the training shapes, and others are the interpolation between those training views from left to right. The first and second training shapes are 12^{o} away from each other along the azimuth angle, and the 2nd and 3rd ones are 10^{o} along the elevation angle.

viewpoints were used, and the interval between each viewpoints are 12^o and 10^o on the azimuth

Figure 6.4: Shape interpolation of one target from each class of targets in each row on the identity manifold. The left, middle and right ones are the training targets, adjacent to each other in each row. Others are the interpolated shape silhouettes between training targets from left to right.

Figure 6.5: Shape interpolation between two adjacent target classes along identity manifold.

and elevation angle direction respectively, which leads to the non-uniformly distributed sampling points on the hemisphere view manifold. Moreover, less training views might be needed as the elevation angle goes larger, this is also due to the less dissimilarity of shape silhouettes along the

view manifold. The way of choosing training viewpoints is closely related to the RBF kernel parameters setting in Eq.4.12 to ensure the validness and effectiveness of the model learning.

We will examine the generative model in terms of its performance and capacity of shape interpolation by three experiments simultaneously:

(1) Shape interpolation along the view manifold by changing the azimuth angle θ and the elevation angle φ respectively. One target is selected from each of the six types of targets and three interpolation image are shown (after thresholding the interpolated distance transformed result) between three training view, as shown in Figure 6.3. A smooth transition along the interpolated shape silhouettes and those training ones can be observed, and this can be more obviously seen from the target wheels transition.

(2) Shape interpolation along the identity manifold within the same class. 3 interpolated shapes along the identity manifold were generated between each of 3 adjacent training targets for each of the 6 target types, as shown in Figure 6.4. We can see that although those training targets are quite bit different from each other, the interpolated shapes preserves the spatial characters from the two adjacent training targets in a very natural way.

(3) Shape interpolation on the identity manifold between two adjacent classes. As shown in Figure 6.5, it is interesting for us to observe the transition of the shape interpolation between two adjacent target classes. Our generative model can create the shape silhouettes smoothly between two adjacent classes and they look realistic, too.

All these result approve to us that the generative model is able to create semantically meaningful shape interpolation along the view and identity manifolds. More importantly, it enables us to analyze a known vehicle from a new given view, but also a unknown target from arbitrary views. Also, the continuous property of both of the manifolds can make the inference process easier.

6.2 Tests on the SENSIAC Dataset

The SENSIAC ATR database contains a large amount of visible and mid-wave IR (MWIR) imagery of seven military and two civilian vehicles. But we will use 8 out of these 9 targets, as shown in Figure 6.6.

Figure 6.6: All eight targets we used for algorithm evaluation from SENSIAC database.

These vehicles are driving along a continuous circle marked on the ground with a diameter of 100 meters (m). These video data were taken at a frame rate of 30 Hz for one minute from distances of 1,000m to 5,000m, (with 500m steps increment between each data) during both daytime and nighttime conditions. In those four methods we discussed earlier in this chapter, we will set $\sigma_v^2 = 1$, $\sigma_x^2 = 0.1$, $\sigma_z^2 = 0.1$, $\sigma_y^2 = 1$ and $\sigma_\varphi^2 = 0.8 \sim 1.2$ due to the fact that vehicles in each videos drives in different speed. But we will make sure that the σ_φ^2 setting is same for all of the four methods within each sequence. We choose 48 different night time IR sequences of eight vehicle targets at six ranges – 1000m, 1500m, 2000m, 2500m, 3000m, and 3500m. Each sequence has approximately 1000 frames. Fortunately each SENSIAC sequence contains a large amount of meta data for each frame. This contains the aspect of the target relative to the sensor and the azimuth of the target relative to the true north, and the elevation of the sensor; the target type, speed, range and slant ranges from the sensor to the target; the pixel location of the target

centroid, and the aspect orientation of the target. Furthermore, we defined a 3D world coordinate system centered at sensor and using a calibration technique based on Pinhole camera to obtain the ground-truth 3D location of the target in each frame as shown in Figure 3.4 and discussed in Chapter3. For your convenience, we show Figure 3.4 here again denoted as Figure 6.7.

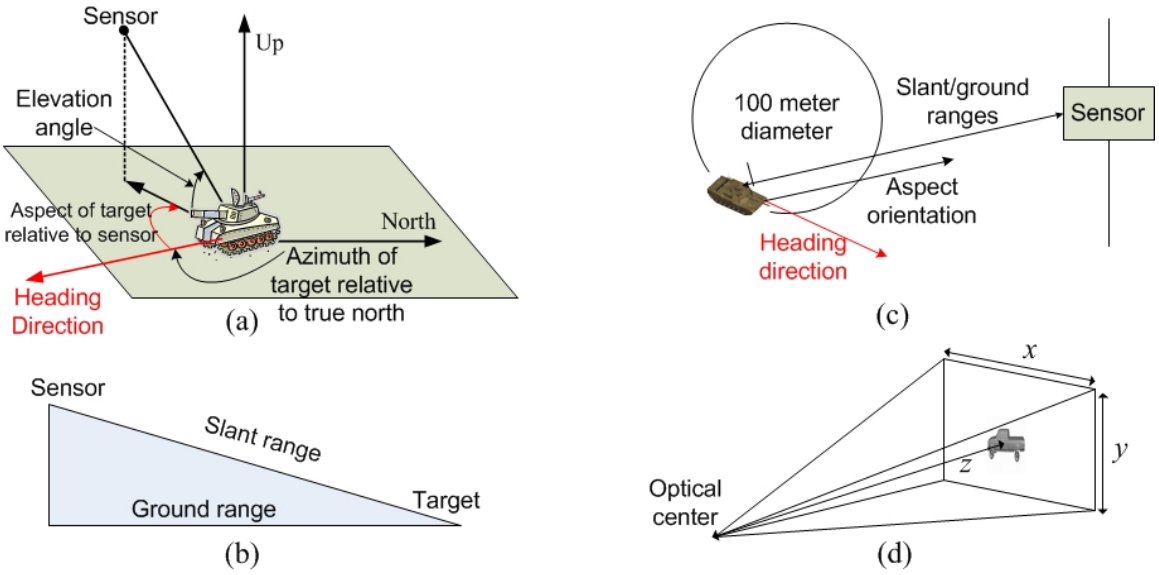


Figure 6.7 (Figure 3.4): Definition of the 3D coordinate system and the spatial geometry of the sensor, and also the target in the SENSIAC dataset. (a) The aspect of the target relative to the sensor, and the azimuth of the target relative to the true north, and the elevation of the sensor; (b) side view of the ground and slant distances between the target and sensor; (c) top-down view of the aspect direction and the heading orientation; (d) A sensor-centered 3D coordinate used for algorithm evaluation.

We initialize the tracking algorithm using ground truth data, including the vehicles' 3D position, and heading direction, and a small positive number for the speed since we don't have ground truth data for this. Another reason is because the data was sampling in a relatively large rate, so between each two frames the distance vehicles traveled was small.

On the identity manifold, the initialization is also using the ground truth, since we know the vehicle's class, and hence we know the true position on the identity manifold. Then random walk was applied on the identity manifold in each tracking frame, at the same time with the particles

applied on the view manifold. The closed –loop structure of the identity manifold prevent any particles- assumptions from going to a dead end on the manifold.

After each tracking frame, the SIR particle filter will give us one state with the largest weight as the current estimation result, and then do the resampling to prepare for the next frame step tracking.

6.2.1 Evaluation of Tracking

The tracking evaluation is based on the errors in approximated 3D target locations along the x (horizontal direction) and z (the range direction) as shown in Figure 6.7 (d). The averaged tracking results over 8 targets with the same range are shown in Figure 6.8. We can see that the errors along the horizontal direction x are all within around 1 meter for all of those four methods. But for the errors on range direction z are much larger than this. It makes sense because the range estimation is relatively much more difficult than the horizontal direction due to the fact that the size of the vehicle is largely related to the range, while the resolution of the imagery and the large distance difference between the circle diameter and the range of target to the sensor is too big, and it is very hard to see the size changes as the vehicle drives as a circle even by human eyes.

Method I shows performance gains of 10%, 20% - 40%, and 3%-50% over Method II, Method III and method IV respectively. Method I also out-performs the other 3 methods on the range and aspect approximation with over 10-50% and 20-80% improvements. These results also show that using the two manifolds together yields the best tracking performance. Even at the range of 3500m, the averaged horizontal, range, and aspect errors of Method I are only 0.5m, 25m and 0.5 rad (28.7^{o}), compared to those result of Method IV's errors of 0.9m, 45m, and 1.1 rad (63.1^{o}). In order to observe the tracking result better, we also show some tracking results for Method I against four 1000m sequences in Figure 6.9, from front view, rear view, and side views. The

interpolated shapes contours are overlaid on the target based on the estimated 3D location and aspect direction and also the given camera calibration. All these results demonstrate effectiveness

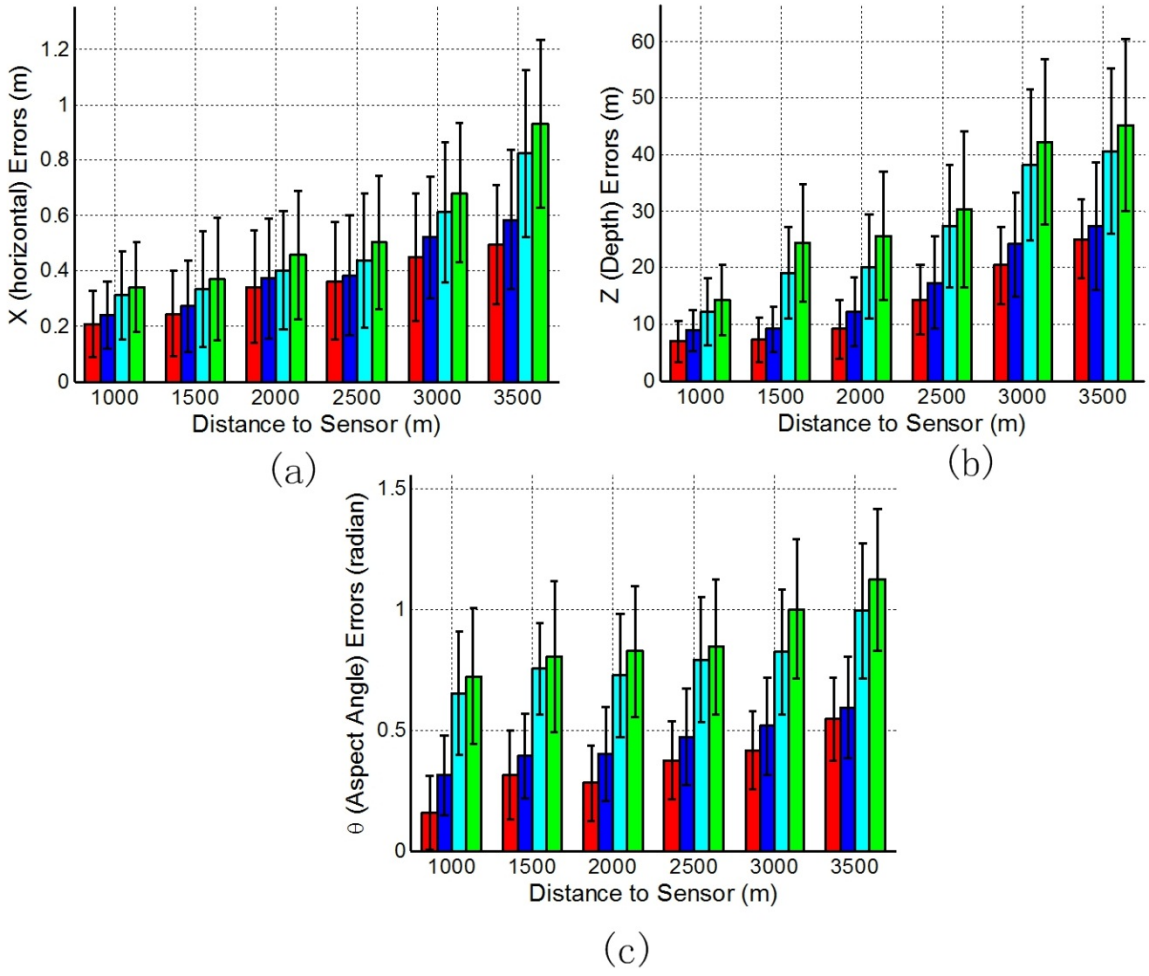


Figure 6.8: Overall 3D tracking evaluation of four different methods. (a) Horizontal error (m); (b) range direction errors (m); (c) aspect angle errors (rad).

and usefulness of the generative model in interpolation target shapes silhouettes along the view and identity manifolds for real ATR applications.

6.2.2 Evaluation of Identity

Since we can map the identity information along the 1D closed loop identity manifold learned from the tensor coefficient space to a 1D unit circle identity manifold, which can significantly simplify our inference procedure, then the identity variable becomes a variable $\alpha \in [0, 2\pi)$ in 1D space. Correspondingly, the six target categories: tanks, APCs, pick-ups, SUVs, mini-vans, and cars, can be represented by 6 angular variables along the circular shaped identity manifold (as shown in Figure 4.4). Since the target type is estimated frame by frame during tracking process,

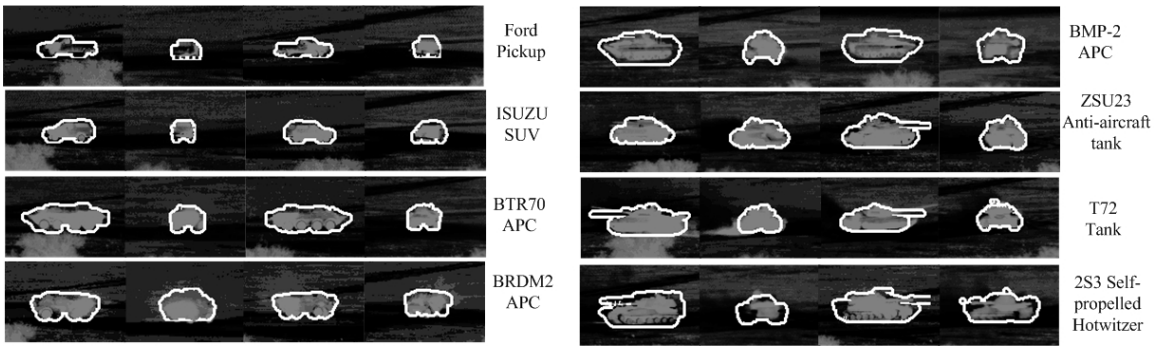


Figure 6.9: Tracking results for eight different targets from SENSIAC database, with original SENSIAC IR imageries overlaid with the interpolated shape contour as white lines.

we define the percentage of frames where the target is correctly recognized in terms of the 6 classes as the overall recognition accuracy, which means the percentage of the angular variable α lies within the range of the true class. And more interestingly, we can also check two training targets that are best matched for a given data that can be found on the identity manifold. The overall recognition performance of the 4 methods for 48 sequences are shown in Table 6.1, where the accuracy of tanks is averaged over those of the T72, ZSU23, and 2S3 and the APCs is averaged over those of BTR70, MP2, and BRDM2. Generally speaking, Method I outperforms other three methods, this further demonstrate the effectiveness of the shape interpolation along both identity and view manifolds. The improvement of Method I is more obvious for long-range sequences when the targets are small and shape interpolation is more significant for better recognition. When the target range is $\geq$2500m, we can see that the recognition result of tanks and

APCs are relatively worse than others, which is below 80%. This is mainly because of the poor segmentation result and small target sizes as shown in Figure 6.1, which shows the segmentation results of all eight targets in all ranges. A simple morphological operation is used to clean up the background subtraction results. However, when the targets are too small, morphological processing has to be adjusted to ensure the target shapes can be preserved better, and this also results in more noisy segmentations. Otherwise, after de-noising, the target shape silhouettes will be seriously damaged, which is impossible for us to do the tracking and recognition.

Targets	Tanks	APCs	SUVs	Pick-ups
1000m	96/94/91/90	94/92/89/88	100/100/99/99	100/100/100/99
1500m	93/91/88/86	88/86/85/82	100/99/98/98	100/100/100/98
2000m	86/83/82/81	85/83/80/80	98/96/96/95	97/96/97/95
2500m	78/73/72/69	76/72/71/70	92/90/89/86	90/88/88/86
3000m	70/65/62/60	72/69/66/65	86/84/82/79	82/80/79/77
3500m	68/62/58/57	70/65/64/62	78/76/75/70	73/72/70/65

Table 6.1: Averaged recognition accuracies (%) of four methods (Method I through Method IV) against forty eight SENSIAC data sequences.

In Figure 6.10, more detailed recognition results of Method I for eight targets under range 1000m are shown. This not only shows the target recognition results frame-by-frame but also the two raining targets that are best matched. I most of the frames, the approximated identity values are in the correct region on the identity manifold and misclassification usually happens at front views and rear views, where the targets are less recognizable and distinguishable. It is interesting that the two best matches of BTR70 and ISUZU-SUV sequences include the exact correct target model. Also, those of other sequences also include a similar target model. For example, BMP1, T72, BRDM1, and AS90 are among the two best matches for BMP2, T80, BRDM2 and 2S3, respectively (both AS90 and 2S3 are self- propelled tanks). Although we do not have the 3D models for the Ford pick-up and the ZSU23 in our training set, their best matches (Chevy/Toyota

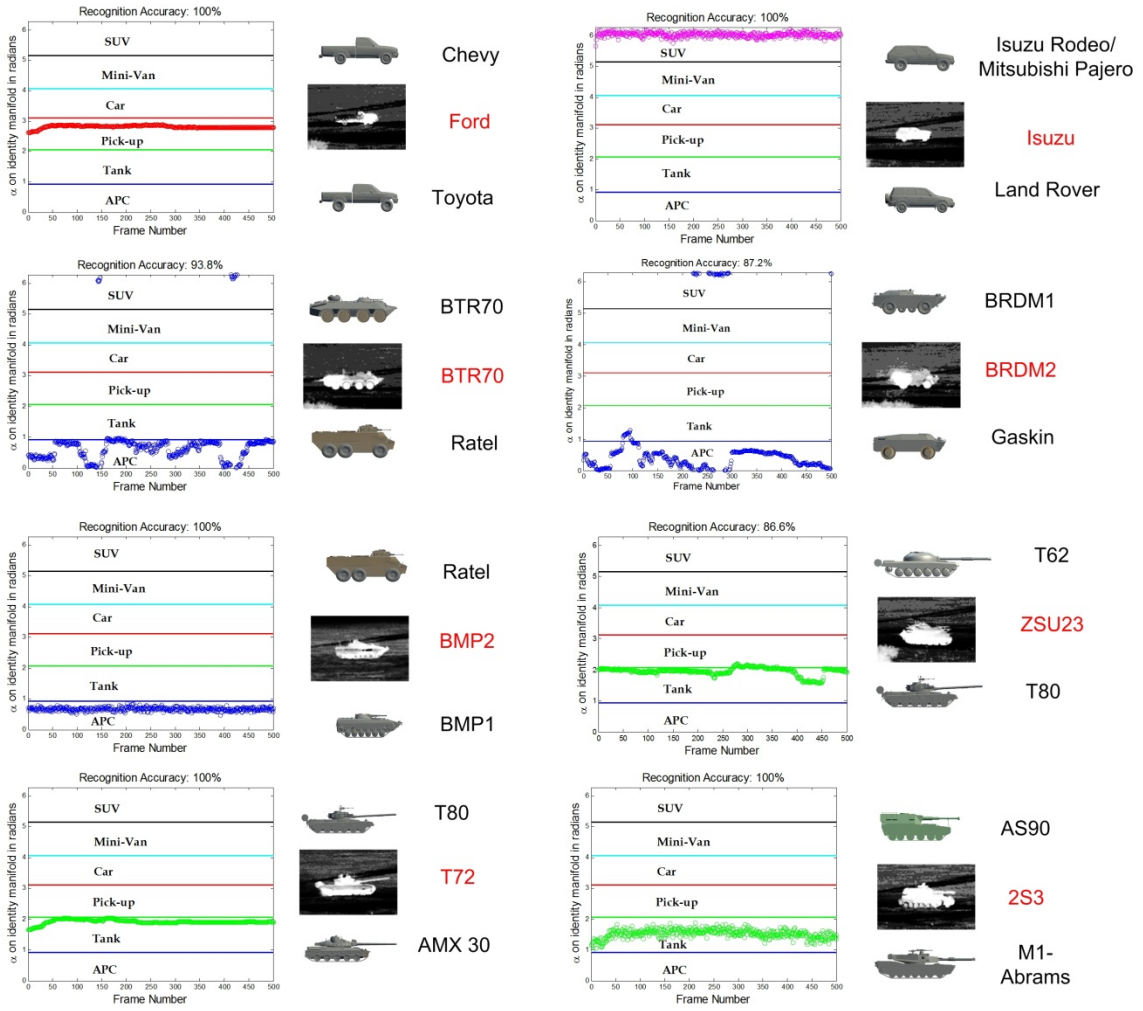


Figure 6.10: Target recognition results frame-wisely shown for eight target sequences at 1000m range distance. For display issue we down-sampled 1000 frames into 500 frames. Vertical axis is the value of the angular value $\alpha \in [0, 2\pi)$ and the range of α corresponding to six different target classes. The estimated target class is also shown along with the two adjacent training target classes.

still resemble the actual targets in the SENSIAC sequences.

6.3 Results on Visible Video Data Sequences

Figure 6.11: Visible data: first row and second row are the data of remote controlled toy vehicle of Lexus IS350 and BMW X5 taken in VCIPL lab respectively; third row is the data from PETS2001 data sequences.

Other than the IR video data sequences from SENSIAC database, we also tested our generative model of four ATR methods on three visible-band video sequences (Figure 6.11). Two of them (data of car and SUV) were captured indoor from a remote controlled toy vehicle, and 3D target tracking and recognition was applied due to the availability of camera calibration for the indoor setting and another one was a real-world surveillance video (a white cargo van) for which camera calibration is not available in the outdoor setting and only pose estimation was performed based on the normalized silhouette sequences. In order to quantitatively compare those four methods, we used an overlap metric [54] to evaluate the overlap between the interpolated shapes with the

segmented targets. Let F and H denotes the tracing gate and the ground-truth bounding box respective, then the overlap ratio ξ is defined as below:

$$\xi = \frac{\#\left(F \text{ I } H\right)\times 2}{\#\left(F\right)+\#\left(H\right)}\ , \tag{6.1}$$

where # is the number of pixels. A larger overlap ratio means larger overlap of the interpolated shapes in the estimated 3D position, and hence better tracking performance. Some examples of this overlap metric are shown in Figure 6.12. Figure 6.13 shows the overlap ratios of four methods on three video sequences. Obviously Method I is much better than other three methods due to it capability of shape interpolation along both view and identity manifolds.

In Figure 6.14 the performance of Method I of three sequences regarding to the recognition are shown. Again, although the models of the three targets are not included in the training data, the recognition accuracy is still 100% for the first two sequences and 97% for the white cargo van, and the two best matches do resemble the unknown target for each sequence. Especially the unknown cargo van is very different from all training models in the class of mini-van, the VW Samba and Nissan Elgrand, which looks much more similar to the cargo van appears in the data gives a reasonable approximation results. The tracking results and pose estimate results are shown

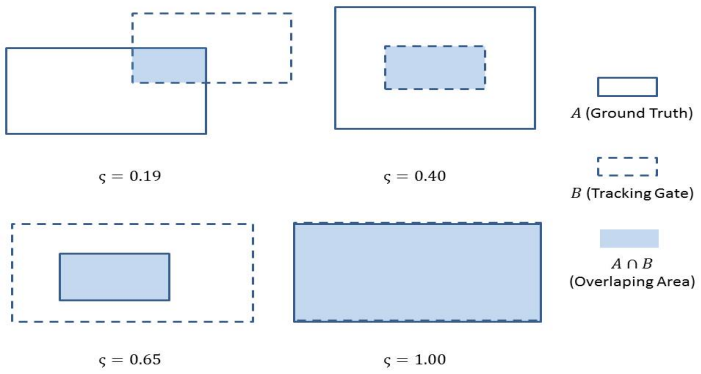


Figure 6.12: Overlap metric for a few example cases

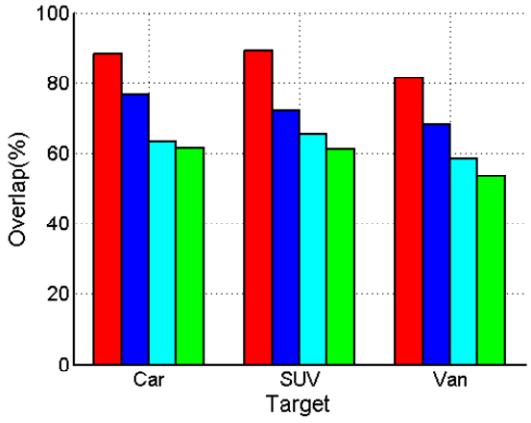
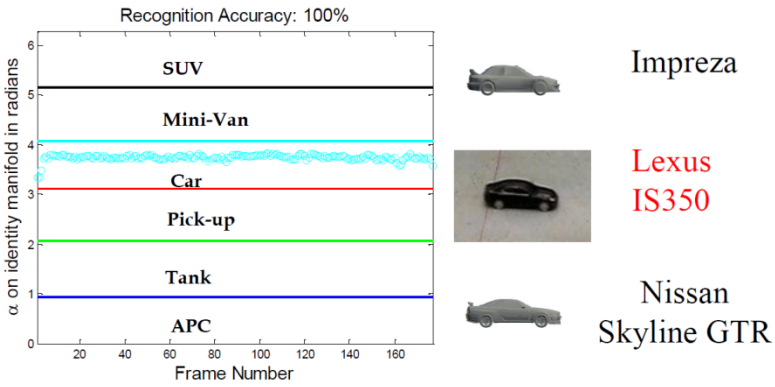
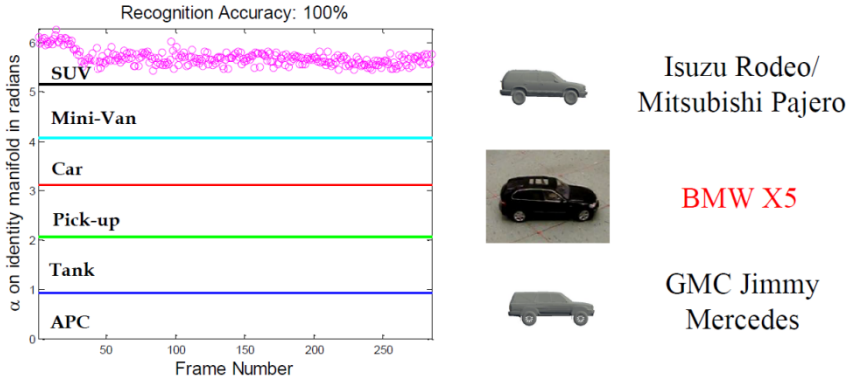


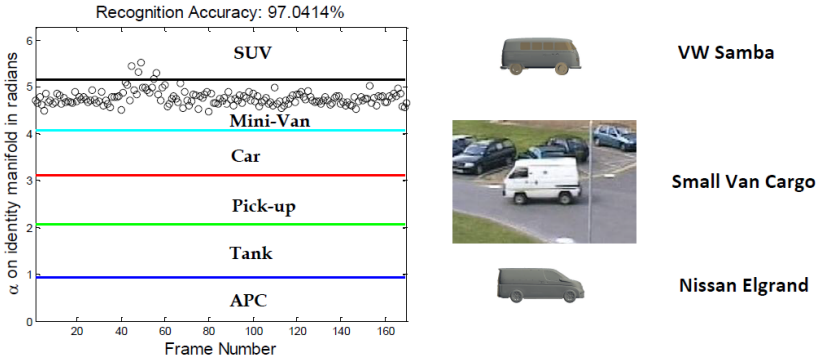
Figure 6.13: Overlap ratios of the target region from segmentation results and synthesized target silhouette for four methods against three real world sequences. From left to right are Method I to Method IV.



(a)

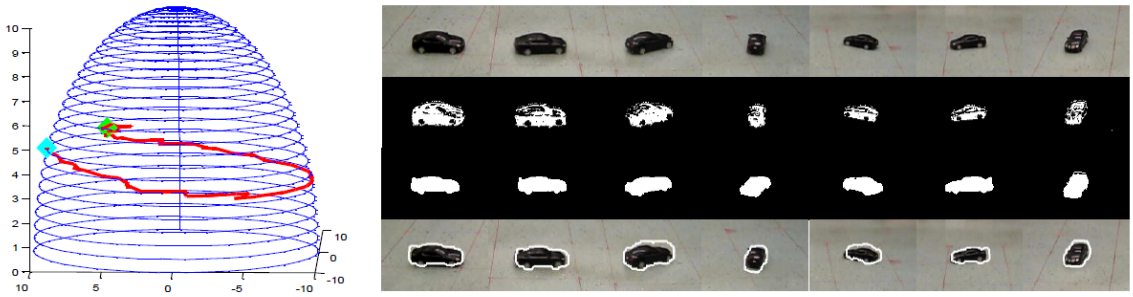


(b)

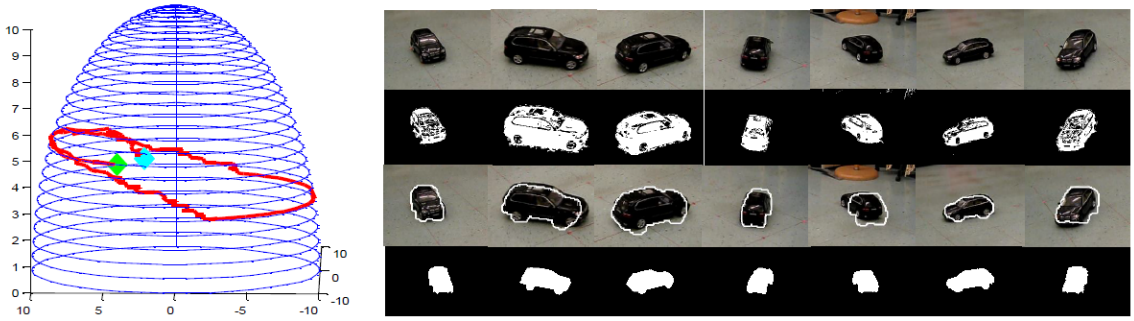


(c)

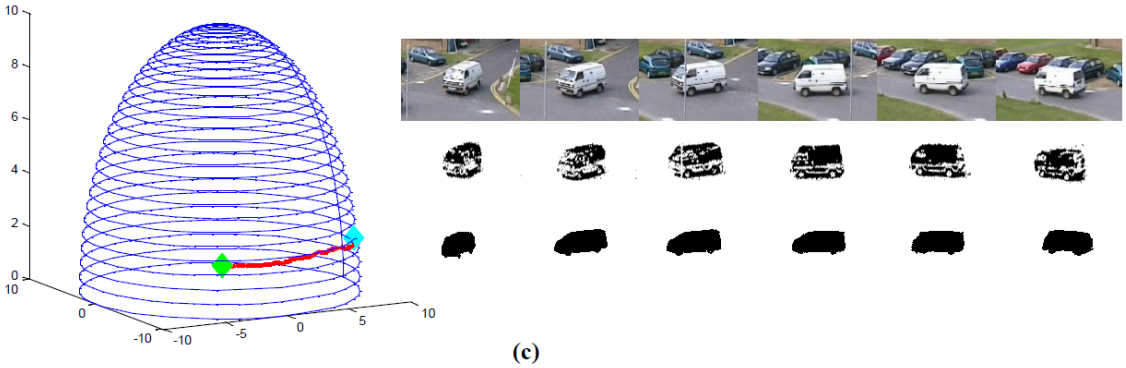
Figure 6.14: Recognition results along with the two best matched training models. (a) car. (b) SUV. (c) mini-van.



(a)



(b)



(c)

Figure 6.15: Tracking results: Left hemisphere and red lines on it show the pose trajectories on the view manifold on the left, and also the selected video frames, segmented objects, and shapes from interpolation and superimposed shapes from the 1st to the 4th rows, while super-imposed result is not available for the cargo van which only pose estimation was performed. (a) Lexus IS350 car. (b) BMW X5 SUV. (c) White cargo van.

in Figure 6.15. In some frames especially in the data of the white cargo van, the segmentation result is relatively worse, but the estimation of the pose trajectories on the view manifold are still smooth and it shows us the real pose changing of the motion target during the sequences were taken. Furthermore, the interpolation results for those three different sequences match well with the segmented target, which indicates that the estimation along both the view and identity manifolds are correct.

6.4 Limitations and Discussion

The generative model is approved to be useful. We can interpolate between each training view and identity, and it can save a lot of memory for us since we do not need to store all the vehicles at each possible training views. This generative model will help us to interpolate semantically meaningful unknown vehicles between training targets under unknown view points, which is not only efficient but also expand our known knowledge of existing targets. But it is still time consuming to learn a model that contains more training target vehicles, and the memory is a main issue during the learning process. Since we need to store all training templates rendered from 3D models, and we need to build a core tensor which has a very large dimension, usually it requires at least 4 gigabytes for learning a model that was used in this work. And the ATR algorithm still cannot be real time processed because of the interpolation and reconstruction of target shapes. So there is still a long way to go in order to further improve the computing efficiency of this generative model.

These experimental results are promising, but we still consider our work preliminary due to two main reasons. First, we use a silhouette-based target shape representation that requires preprocessing of the target segmentation. It is fine to assume that the camera platform is static with static camera. But in the case of a moving camera system, the preprocessing of the target segmentation will be a very challenging issue. Second, we did not consider the occlusion issue

that has to be handled with in any real ATR systems. The silhouette could be very sensitive to those occlusion issues. So we could extend our algorithm to some other features such as SIFT or HOG features which are much more salient and stable. By doing this we can definitely increase the ability of the application of the generative model for real world applications.

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

In this work we further expand the application of a recently presented generative model which combines a continuous identity manifold for target recognition and also a multi-view shape-based generative model which combines a hemisphere view manifold. We include military tanks and APCs in our training dataset, together with civilian cars, pick-ups, SUVs and mini-vans. And hence we apply this generative model to the ATR application using IR imagery from SENSIAC database, and also some visible-band sequences. We proposed a particle filter-based inference algorithm that involves tracking and recognition at the same time during inference process. The experiments show us the advantages of shape interpolation along both the view and identity manifold, and the tracking and recognition accuracies using this generative model always outperform other three comparison methods.

However, since we are using shape information as the target appearance, and background subtraction is usually a hard issue. Moreover, in IR data the shape information is always not accurate compared with visible-band imagery data. Most importantly, the target shape information (i.e. silhouette) is relatively sensitive if occlusion is counted. Due to these facts, we propose to learn this generative model by using some features like SIFT, and HOG feature. These features can strongly capture the appearance and identity of the interested target, and also less sensitive when occlusion happens.

Furthermore, factorizing out the identity from view information is based on the assumption that view and identity is independent from each other. But in the real world they are always influence by each other to some extent. For example the shape of a ball-shaped target will not change from different view point, but a cube-shaped target will have significant changes. Based on these factors, we propose to learn a generative model that not only combining the view and identity manifolds, but also incorporate them together into one generative manifold that captures the view information based on identity at the same time, and hence apply it to the IR imagery based ATR applications.

REFERENCES

1. K. Mikolajczyk, Multiple Object Class Detection with a Generative Model. *in Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2006.

2. Y. Li, L. Gu, and T. Kanade. A robust shape model for multi-view car alignment. in proc. *IEEE Conference on Computer Vision and Pattern Recognition,* 2009.

3. J. Liebelt and C. Schmid. Multi-view object class detection with a 3D geometric model. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2010.

4. V. Venkataraman, X. Fan, and G. Fan. Integrated target tracking and recognition via joint appearance-motion generative models. in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops,* 2008.

5. M.I. Miller, et al., Automatic target recognition organized via jump-diffusion algorithms. *IEEE Trans. on Image Processing,* 1997. 6(1): p. 157-174.

6. X. Mei, S.K. Zhou, and W. Hao. Integrated Detection, Tracking and Recognition for IR Video-Based Vehicle Classification. in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing,* 2006.

7. J.S. Shaik and K.M. Iftekharuddin. Automated tracking and classification of infrared images. in *Proc. of the International Joint Conference on.Neural Networks,* 2003.

8. V. Venkataraman, G. Fan, and X. Fan. Target Tracking with Online Feature Selection in FLIR Imagery. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition,* 2007.

9. V. Venkataraman, et al. Appearance learning by adaptive Kalman filters for FLIR tracking. in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops,* 2009.

10. Z. Zhang, et al. EDA Approach for Model Based Localization and Recognition of Vehicles. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition,* 2007.

11. L.A. Chan and N.M. Nasrabadi. Modular wavelet-based vector quantization for automatic target recognition. in *IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems,* 1996.

12. L. Wang, S.Z. Der, and N.M. Nasrabadi, Automatic target recognition using a feature-decomposition and data-decomposition modular neural network. *IEEE Trans. on Image Processing*, 1998. **7**(8): p. 1113-1121.

13. Military Sensing Information Analysis Center (SENSIAC). https://www.sensiac.org/, 2008.

14. PETS 2001. http://www.cvg.cs.rdg.ac.uk/PETS2001/pets2001-dataset.html.

15. T. Poggio and S. Edelman, A network that learns to recognize three-dimensional objects. *Nature*, 1990. 343(6255): p. 263-266.

16. Ullman, S. and R. Basri, Recognition by linear combinations of models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1991. 13(10): p. 992-1006.

17. S. Ullman and R. Basri, An approach to object recognition: Aligning pictorial descriptions. *Cognition*, 1989. 32:193-254.

18. O.C. Ozcanli, et al. Augmenting Shape with Appearance in Vehicle Category Recognition. in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

19. R. Savarese and Fei-Fei L., Multi-view Object Categorization and Pose Estimation, in *Computer Vision*, Volume 285 of studies in Computational Intelligence. Springer, 2010.

20. H. Su, et al. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. in *Proc. IEEE 12th International Conference on Computer Vision*, 2009.

21. A. Toshev, A. Makadia, and K. Daniilidis. Shape-based object recognition in videos using 3D synthetic object models. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

22. A. Kushal, C. Schmid, and J. Ponce. Flexible Object Models for Category-Level 3D Object Recognition. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

23. S.M. Khan, et al. 3D model based vehicle classification in aerial imagery. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

24. M.J. Leotta, and J.L. Mundy. Predicting high resolution image edges with a generic, adaptive, 3-D vehicle model. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

25. J. Luo, et al. 3-D model based vehicle tracking. IEEE Trans. Image Processing, 2005. 14:1561-1569.

26. R. Sandhu, et al. Non-rigid 2D-3D pose estimation and 2D image segmentation. in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

27. Y. Tsin, Y. Gene, and V. Ramesh. Explicit 3D Modeling for Vehicle Monitoring in Non-overlapping Cameras. in *Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2009.

28. Dian-yong, Z. and M. Zhen-jiang. 3D Human shape reconstruction from photographs based template model. in *Signal Processing, 2008. ICSP 2008. 9th International Conference on*. 2008.

29. Pingkun, Y., S.M. Khan, and M. Shah. 3D Model based Object Class Detection in An Arbitrary View. in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. 2007.

30. M.F. Abdelkader, et al., Silhouette-based gesture and action recognition via modeling trajectories on Riemannian shape manifolds. Comput. Vis. Image Underst., 2011. 115(3): p. 439-455.

31. S. Belongie, J. Malik, and J. Puzicha, Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2002. 24(4): p. 509-522.

32. M. Hu, Visual pattern recognition by moment invariants. *IEEE Trans. on Information Theory* , 1962. 8(2): p. 179-187.

33. A. Elgammal and L. Chan-Su. Separating style and content on a nonlinear manifold. in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

34. Chan-Su, L. and A. Elgammal. Modeling View and Posture Manifolds for Tracking. in *Proc. IEEE 11th International Conference on Computer Vision*, 2007.

35. A. Srivastava, et al. Statistical shape analysis: clustering, learning, and testing. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2005. 27(4): p. 590-602.

36. H. Murase and S.K. Nayar, Visual learning and recognition of 3-D objects from appearance. *Int. J. Comput. Vision*, 1995. 14(1): p. 5-24.

37. J.B. Tenenbaum and W.T. Freeman, Separating Style and Content with Bilinear Models. *Neural Comput.*, 2000. 12(6): p. 1247-1283.

38. M.A.O. Vasilescu and D. Terzopoulos, Multilinear Analysis of Image Ensembles: TensorFaces, in *Procs. of the 7th European Conference on Computer Vision-Part I*. 2002, Springer-Verlag. p. 447-460.

39. C. Gosch, et al. View Point Tracking of Rigid Objects Based on Shape Sub-manifolds, in *Proceedings of the 10th European Conference on Computer Vision: Part III*. 2008, Springer-Verlag: Marseille, France. p. 251-263.

40. Cheng-Hao, K. and R. Nevatia. Robust multi-view car detection using unsupervised sub-categorization. in *Applications of Computer Vision (WACV), 2009 Workshop on*. 2009.

41. H.H. Bulthoff and S. Edelman, Psychophysical support for a two-dimensional view interpolation theory of object recognition. *Proc Natl Acad Sci U S A*, 1992. 89(1): p. 60-4.

42. S. Cheung and C. Kamath, Robust background subtraction with foreground validation for urban traffic video. *EURASIP J. Appl. Signal Process.*, 2005. 2005: p. 2330-2340.

43. K. Toyama, et al. Wallflower: principles and practice of background maintenance. in *Proc. Seventh IEEE International Conference on Computer Vision*, 1999.

44. J. Carranza, et al., Free-viewpoint video of human actors, in *ACM SIGGRAPH 2003 Papers*. 2003, ACM: San Diego, California. p. 569-577.

45. I. Miki, et al. Human Body Model Acquisition and Tracking Using Voxel Data. *Int. J. Comput. Vision*, 2003. 53(3): p. 199-223.

46. C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999.

47. Z. Zivkovic and F.v.d. Heijden, Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recogn. Lett.*, 2006. 27(7): p. 773-780.

48. Camera Calibration Toolbox for MATLAB. http://www.vision.caltech.edu/bouguetj/calib_doc/.

49. R. Chellappa, C.L. Wilson, and S. Sirohey, Human and machine recognition of faces: a survey. *Proceedings of the IEEE*, 1995. 83(5): p. 705-741.

50. T. Kolda and B. Bader, Tensor Decompositions and Applications. *SIAM Review*, 2009. 51(3): p. 455-500.

51. M.A.O. Vasilescu and D. Terzopoulos. *Multilinear image analysis for facial recognition.* in *Proc. 16th International Conference on Pattern Recognition*, 2002.

52. Rong Li, X. and V.P. Jilkov, Survey of maneuvering target tracking. Part I. Dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 2003. 39(4): p. 1333-1364.

53. M.S. Arulampalam, et al., A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions. Signal Processing*, 2002. 50(2): p. 174-188.

54. S. Kai, et al. Vehicle tracking using on-line fusion of color and shape features. in *Proc. The 7th International IEEE Conference on Intelligent Transportation Systems*, 2004.

VITA

Liangjiang Yu

Candidate for the Degree of

Master of Science

Thesis: JOINT TARGET TRACKING AND RECOGNITION USING SHAPE-BASED GENERATIVE MODEL

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Bachelor of Science in Communication Engineering at North University of China, Taiyuan, Shanxi, China in 2008.

Experience:

Visual Computing and Image Processing Lab (VCIPL), OSU, Stillwater, OK
Research Assistant, Aug.2009 – Present

Professional Memberships:

IEEE Student Membership

Name: Liangjiang Yu Date of Degree: December, 2011

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: JOINT TARGET TRACKING AND RECOGNITION USING SHAPE-BASED GENERATIVE MODEL

Pages in Study: 69 Candidate for the Degree of Master of Science

Major Field: Electrical Engineering

Scope and Method of Study:

Recently a generative model that combines both of identity and view manifolds was proposed for multi-view shape modeling that was originally used for pose estimation and recognition of civilian vehicles from image sequences. In this thesis, we extend this model to both civilian and military vehicles, and examine its effectiveness for real-world automated target tracking and recognition (ATR) applications in both infrared and visible image sequences. A particle filter-based ATR algorithm is introduced where the generative model is used for shape interpolation along both the view and identity manifolds. The ATR algorithm is tested on the newly released SENSIAC (Military Sensing Information Analysis Center) infrared database along with some visible-band image sequences. Overall tracking and recognition performance is evaluated in terms of the accuracy of 3D position/pose estimation and target classification.

Findings and Conclusions:

The generative model that is learned from both civilian and military vehicles shows semantically meaningful shape interpolation along both the view and identity manifolds, showing its capability to handle both known or new targets under arbitrary views. Comparing with three baseline algorithms with limited or no shape interpolation, the new ATR algorithm demonstrates the advantages of using the generative model to support more accurate 3D tracking, pose estimation and target classification.

ADVISER'S APPROVAL: Dr. Guoliang Fan