ARTICULATED HUMAN MOVEMENTS TRACKING THROUGH ONLINE DISCRIMINATIVE LEARNING

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Kyuseo Han

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

May 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Avinash C. Kak, Chair

School of Electrical and Computer Engineering

Dr. Hong Z. Tan

School of Electrical and Computer Engineering

Dr. Fengqing M. Zhu

School of Electrical and Computer Engineering

Dr. Tanmay Prakash

School of Electrical and Computer Engineering

Approved by:

Dr. Dimitrios Peroulis

Head of the School of Electrical and Computer Engineering

ACKNOWLEDGMENTS

Sujin, my lovely wife, undoubtedly is the most deserved one to have the greatest appreciations. During this time, she has always been my first and the most sincere supporter. Her incredible sacrifice and assistance realized all thing I made in my study. Without her persistent supports, I could not have reached the finish line of this journey.

I wish to give another big appreciation to my parent who have continuously stood beside me and showed their love.

I would like to express my sincere and wholeheartedly appreciation to my supervisor, Professor Avinash Kak, who convincingly guided me to walk on the right path in the dark woods. His motivation, passion, and immense knowledge encouraged me to expand my research perspectives.

Besides my supervisor, I wish to show my deepest gratitude to all of my committee members, Professor Hong Tan, Professor Fengquin Zhu, and Dr. Tanmay Prakash, for their invaluable comments and assistance.

Last but not least, I give all my thanks to the Lord allowing me to take this path.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

ABSTRACT

Han, Kyuseo Ph.D., Purdue University, May 2020. Articulated Human Movements Tracking through Online Discriminative Learning. Major Professor: Avinash C. Kak Professor.

In this thesis, we present a new class of object trackers that are based on a boosted Multiple Instance Learning (MIL) algorithm to track an object in a video sequence. We show how the scope of such trackers can be expanded to the tracking of articulated movements by humans that frequently result in large frame-to-frame variations in the appearance of what needs to be tracked. To deal with the problems caused by such variations, we present a component-based MIL (CMIL) algorithm with boosted learning. The components are the output of an image segmentation algorithm and give the boosted MIL the additional degrees of freedom that it needs in order to deal with the large frame-to-frame variations associated with articulated movements. Furthermore we explored two enhancements of the basic CMIL tracking algorithm. The first is based on an extended definition of positive learning samples for CMIL tracking. This extended definition can filter out false-positive learning samples in order to increase the robustness of CMIL tracking. The second enhancement is based on a combined motion prediction framework with the basic CMIL tracking for resolving issues arising from large and rapid translational human movements. The need for appropriate motion transition can be satisfied by probabilistic modeling of motion. Experimental results show that the proposed approaches yield robust tracking performances in various tracking environments, such as articulate human movements as well as ground human movements observed from aerial vehicles.

1. INTRODUCTION

1.1 Problem Statement

Human tracking has been a key issue in several computer vision applications that include pedestrian safety, surveillance, crowd analysis, border control, sports broadcasting, and so on. For example, the retail sector benefits from customers' movement trajectories in the stores for recognizing and analyzing customers' activities because the customers' typical activities in stores are significant resources for retailers to adequately arrange the goods and shelves. This allows them to provide customers with convenient shopping experiences as well as increase their revenues. In border control, it is essential to detect and track human and vehicle movement for observing illegal immigration attempts across border.

Traditionally, there are two basic approaches to human tracking: an appearance-based approach and a structured-body-part-based approach. The appearance-based approach needs a reference template of a human of interest to be tracked that can usually be defined by specific feature distributions in a bounding box that surrounds the human to be tracked. The goal of appearance-based tracking is to find the local region in the given input frame that has the most similar feature distributions to those of the reference template. Note that the feature distributions have been expressed with specific features, e.g., color, edge, gradient orientation, etc. Unfortunately, the appearance-based approach often suffers from temporal variations in feature distributions due to clutter backgrounds, varying illuminations, or partial occlusions. Consequently, it is necessary to select appropriate features that can produce robust distributions for the temporal variations in order to guarantee better performance for the appearance-based tracking.

The structured-body-part-based approach is an alternative way of human tracking that can compensate for the vulnerability of the appearance-based tracking. The structured-body-part-based approach depends on joint configuration models for human body parts, e.g., head, arm, torso, leg, etc. The joint configuration model can specify the feasible connections of one human body part with another. For example, the human head can have a connection with the torso but cannot have one with the legs. The structure-body-part-based approach can be robust to even some occlusions because the predefined joint configuration model can provide possible connections of human body parts. To achieve robustness, the structure-body-part-based approach necessarily requires a high quality of human part detection as well as predefined joint configuration models.

Both appearance- and structured-body-part-based approaches have reported good tracking performance for simple human movements, e.g., walking and running. However, neither of appearance- and structured-body-part-based tracking approaches can appropriately handle the extensively articulated human movements, e.g., bending, crunching, jumping, etc. The reference template in the appearance-based tracking approach has been contaminated by severe appearance changes in extensively articulated human movements. On the other hand, the structured-body-part-based tracking approach has been impaired by failures of human parts detection followed by unpredictable results due to mismatched body part connections.

The unsuccessful conduct of both tracking approaches with extensively articulated human movements is caused by the non-rigid shape of humans themselves. The conventional object tracking techniques assume that the object to be tracked is a rigid one – its appearance can hardly be changed during tracking – therefore, the center-of-mass position of the rigid object is an actual value for a target to be tracked. In contrast, the human body is so flexible and pliable that object rigidity is not an adequate assumption for tracking humans; especially large articulated movements, such as bending, crunching, rolling etc., can deteriorate the human tracking performance.

Successful tracking for extensively articulated human movements ensures that the human tracking techniques take into special consideration the non-rigid shape of the human body.

1.2 A Tracking Framework with Classification

The recent computational developments and advances have shed a light on real-time tracking frameworks based on classification techniques. Specifically, the tracking-by-detection framework has been widely promoted for the online classification-based tracking approaches. In the tracking-by-detection framework, classifiers can determine the best estimated position, where the object to be tracked is likely located, among various candidate positions in an input image. The classifier for tracking-by-detection has usually been composed of a binary classification (two-class classification) rather than a multiple-class classification because the binary classifier can conceptually separate the object to be tracked and the background in the given image. The binary classification naturally requires positive and negative training samples for training classifiers: the positive training samples and the negative ones can be obtained from the object to be tracked and the background, respectively. Ideally, the optimal classifiers can be computed from a complete set of both positive and negative training samples. However, the classifier for tracking-by-detection for an online learning-based tracking should be inherently trained from each input image during tracking. In other words, the classifiers can be incrementally updated with subsets of training samples that are collected in successive input images: the classifier can be updated by extracting information from the best estimated position of the object to be tracked in the current input image. Therefore, it is necessary to obtain as evenly distributed sampled subsets of both positive and negative training samples on the entire sample space as possible from each image. In real environments, however, most online classifications have suffered from the imprecise sampling of training samples

and outliers due to being impractical to collect the evenly distributed sampled subset of training samples in each frame.

1.3 Online classification for articulated human tracking

This thesis primarily explores how to track the extensively articulated human movements via the online classification technique. Specifically, we develop a more robust way of updating the reference appearance template, which cooperates with the human part joint configuration. The updated reference appearance template alleviates failures of tracking due to significant appearance changes and severe human pose changes during tracking the human. The online classification technique efficiently accomplishes the adaptive update of the reference appearance template for dealing with the extensively articulated human movements.

In this thesis, Multiple Instance Learning (MIL) has been adapted as the online classification for the tracking framework. MIL is a variant of the supervised learning approach with special managements of its training samples. Instead of using each individual training sample, MIL utilizes a set of training samples, what we call a "bag". Hereafter, the training sample in MIL is the bag, and an element of each bag is an instance. The positive bag is defined as the bag with at least one positive instance; it means that the positive bag can contain both positive and negative instances. The negative bag, on the contrary, should contain only negative instances. For successful tracking of articulated human movements with online classification, it is significant to collect true positive training samples in each input image because the selected positive training samples likely include false positive training samples due to severe changes in successive input images. Therefore, there exist some ambiguities in both positive and negative training samples. MIL can handle this ambiguity in its inherent training stage, which can minimize incorrect classification due to ambiguous training samples.

This work, an online MIL learning and classification for articulated human movements, focuses on two main objectives: how to generate the reference template while taking into consideration human part configurations and how to collect positive training samples more confidently.

First, the generation of the reference template of humans to be tracked is achieved by segmenting the template into several small regions, what we call "components." Each component characterizes human body parts because components in the reference template can be segmented with respect to local similarity. The components in the reference template can be positive instances in the context of MIL. Furthermore, the component-based MIL(CMIL) is extended to use the adaptively configured positive bag. The extension of positive bag can be declared when at least the minimum number of positive components exist in the bag. The minimum number of positive components can also be adaptively changed so that positive bags can be configured while taking into account appearance changes during tracking due to illumination changes, severe human movements, background changes, partial occlusions, etc. The updated reference template can be configured by more than one positive component to compensate for these appearance changes.

Second, the selection of true positive training samples can be achieved by minimizing chances to select a false positive training sample. The adaptive selection of positive training samples can decrease the opportunity to select false positive training samples. The conventional random or uniform sampling of positive training samples within a search range easily generates false positive samples due to severe shape or appearance changes. These false positive samples aggravate the locally optimized classifiers due to the limitation of partial sets of training samples. In this thesis, the probable sampling positive bags for online MIL can provide better positive training sample selections. The aggregation with the particle filter framework provides the probabilistic estimation of positive training samples. This probabilistic sampling contributes to minimizing the chances of selecting false positive learning samples for

online CMIL and provides better tracking performance when large translation motions with articulated movements exist.

1.4 Outlines

In Chapter 2, we introduce some brief background of human tracking and MIL. Chapter 3 explains the concept of component-based MIL and some preliminary experiments for extensive articulated human movement tracking. In Chapter 4, the adaptively configured positive sample is introduced for reducing false positive samples in component-based MIL tracking. In Chapter 5, we also propose to transfer good positive samples over the successive video frames via Particle Filter technique. We conduct a comparative study with CNN based tracking algorithms in Chapter 6. Chapter 7 illustrates how the MIL with Particle filter could be applied to Unmanned Aerial Vehicle systems for tracking humans on the ground. Finally, we conclude the component-based MIL for extensive articulated human movement tracking in Chapter 8.

2. OVERVIEW OF TRACKING HUMAN MOVEMENTS WITH ONLINE LEARNING

This chapter provides an overview for tracking articulated human movements through the tracking-by-detection paradigm, which is specifically the online learning system. First, we explain how to track human motions through two conventional approaches: (1) the appearance-based approach and (2) the structured-body-part-based configuration approach. Moreover, we summarize existing problems in articulated human movements tracking. Second, as the tracking-by-detection method, Multiple Instance Learning (MIL) will be introduced for alleviating existing problems in articulated human movement tracking. MIL can provide a learning strategy that secures good performance of classifiers for detecting even the humans who have severe changes of appearance and shape. Finally, online Adaboost will be introduced for learning classifiers in order to track human movements. The online learning mechanism is inevitably required to track humans in a given video sequence in order to support on-the-fly operations of human tracking.

2.1 Human Motion Tracking

Tracking humans in motion is an important part of computer vision in applications that involve human subjects. Despite the fact that there now exist several algorithms for this purpose, the solutions obtained with the current algorithms are generally unsatisfactory in real-world applications [1].

For the problem of tracking humans when they are engaged in ordinary movements — such as walking — the solutions developed fall into two categories: those that are based on bounding boxes and those that are not. The bounding-box-based solutions track the image intensity distributions within a bounding box placed around the

target in the first image of a sequence and then use either a Kalman filter [2, 3] or a particle filter [4, 5] to track the distribution. On the other hand, the algorithms that do not require a bounding box are based on using a probabilistic framework that can jointly carry out the segmentation that best describes the target in each frame and tracking [6, 7]. For tracking more complex human movements, researchers have proposed methods that use a parts-based model, with the parts standing for head, torso, arm, leg, belly, etc. [8,9]. A major shortcoming of these methods is the need to identify the parts in the images. In some cases, one can get around this shortcoming by using a probabilistic framework [10–12]. However, even these approaches break down when the extent of self-occlusion is significant. To deal with the problems caused by occlusion, researchers have proposed tracking methods based on machine learning algorithms, such as a cascade of SVM classifiers [13], boosted cluster tree [14], etc.

The two main categories for human tracking are described as: (1) appearance-based tracking and (2) human-body-part-based tracking. Figure 2.1 and Figure 2.2 show the concepts of both appearance-based tracking and human-body-part-based tracking, respectively.

Appearance-based tracking has been widely used for tracking objects and humans in computer vision. The appearance model is usually represented by an intensity distribution within a bounding box that probably surrounds the human subject. Consequently, the goal of human tracking based on the appearance model is to find the position of the bounding box at which the true intensity distribution of the human subject is located over successive image sequences. Either a Kalman filter or a particle filter have conventionally been used for tracking the intensity distribution over the video sequence [2–4,15]. The Kalman filter has traditionally been used for object tracking and, of course, for human tracking. Bertozzi et al. [2] use Kalman filter to track individual pedestrians with merging overlapped region of interest (ROI). The individual pedestrian is initially extracted from utilizing edge symmetry of the detected object with stereo refinement. Grubb et al. [3] use the Kalman filter for tracking the spatial information of pedestrians with Bayesian probability for tempo-

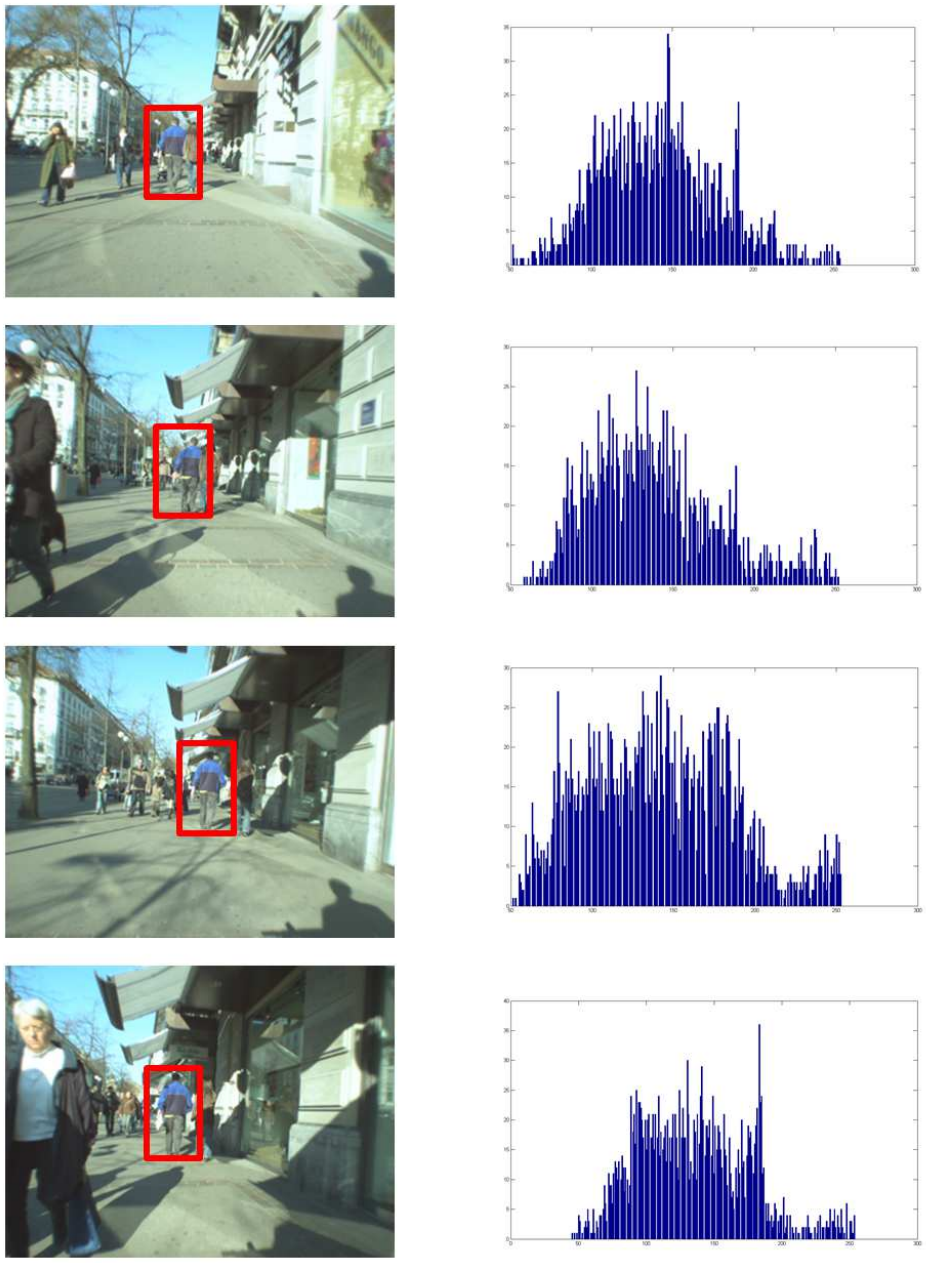
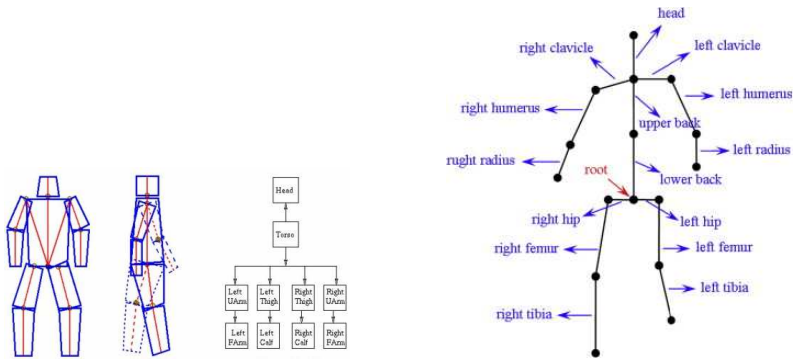


Fig. 2.1. Example of appearance model for human tracking. The intensity distribution can be expressed by histogram

ral update. Also, Support Vector Machine (SVM) is used for pedestrian detection with the extended Kalman filter for human tracking [16]. The initial candidates are extracted from stereo vision and refined by SVM.

Particle filter is also widely used for human tracking. Giebel et al. [4] proposed the particle filter based 3D object tracking with integrating multiple-cues, such as shape, texture, and stereo cues. Similar to the extended Kalman filter with SVM, a variant of SVM with the particle filter is used for human detection [5]. Tuong et al. [15] also proposed the particle filter based pedestrian detection and tracking with HoG descriptor.

Both Kalman filter and particle filter tracking usually estimate the center position of the bounding box that encapsulates the human subject with conventionally simple human poses, for example the walking and standing of pedestrians. Specifically both Kalman filter and particle filter based tracking frameworks have inherent limitations in their ability to track large articulated movements on account of the fact that the first-order probability distributions for the pixel intensities inside the object being tracked undergo both large variations and large rates of variations that are beyond the capabilities of such trackers. The presence of severe occlusion, complex background clutter, and large illumination changes hinder these appearance-based trackers because the intensity distribution in the bounding box, which probably encapsulates the human subject being tracked, is severely distorted from the true intensity distribution.



(a) (b)

Fig. 2.2. Example of human body part configurations: (a) Human body topology (www.ifp.illinois.edu/˜yuhuang/Body.html) and (b) Human body skeleton configuration [17].

An alternative approach to human tracking is known as the human-body-parts-based approach that has gained popularity for overcoming some deficiencies of the appearance-based trackers. When such trackers are used for tracking human movements, they seek to find several regions that correspond to the parts of the human body, such as head, torso, arms, and legs, followed by merging these regions into one big region which contains the human subject being tracked by pre-defined configurations that describe how to build the human body with separate body parts. In any configurations, the fact of restricted connections among human body parts, e.g., the head can be connected with the torso but not with the legs, can improve the performance of both the detection of the parts and the assignment of labels to the parts. Using these ideas, some human-body-parts-based trackers have been proposed [8, 10–12, 18] for human tracking and multi-person tracking. Andriluka et al. [10] proposed an articulation and dynamic limb-based detector using a hierarchical Gaussian latent model. Dollar et al. [18] proposed the part-based learning system for object tracking. They proposed the multiple component learning, which is similar to multiple instance learning; each component represents a part of the object to be tracked with a restricted assumption that the component appears in a specific region

of the image. Multi-person tracking has been proposed with considering body poses for an improved observation model [8]. Each articulated tracker extracts pedestrian silhouettes via segmentation and provides pose reconstruction and shape prediction based on PCA with combining particle filter. Lin et al. [12] suggested the part-based object detection algorithm. The object of interest consists of several parts; each part has no semantics and its own positive and negative instances represented by local image regions. Felzenszwalb et. al [19] present a mixture of the deformable part model for object detection that employs a global template and deformable part. Both are represented by HOG feature templates that are trained by latent SVM. Breitenstein et al. [11] proposed online multi-person tracking with tracking-by-detection. Each detector provides its confidence, which is used for updating both corresponding trackers with implementation by a particle filter and a classifier handling false positive detections as well as false negative detections.

While the human-body-parts-based trackers are superior to the appearance-based trackers in tracking articulated human movements, there is an upper bound on the extent of articulation they can tolerate. For example, they usually fail when the human movements include bending, crunching, or kneeling: such movements create a whole alteration of the configuration that defines the relationship between human body parts. The configuration changes entailed by such movements can be large enough to create new adjacency and connectedness between the parts while destroying those that applied at the start of the movement. This limitation of parts-based trackers was the focus of a recent contribution by Zhang et al. [13]. Their framework includes a detector that is based on a cascade of SVM classifiers. The tracking itself in their work is driven by a simple silhouette-based model for the foreground and the tracking based on a sliding window is executed by associating local trajectories from MAP. A different approach was taken by Yang et. al [14] where the authors have exploited edgelet features for detecting articulated variations in human poses caused by standing, bending, crouching, and sitting. They heuristically optimized

the sample split strategy; the samples are split into different clusters so that each cluster includes as similar of a sample as possible.

2.2 Learning based human pose estimation

The recent burst of computing power has realized complex machine learning techniques that had not been viable due to a lack of computational capacity and memory storage. One of the beneficiaries is a convolution neural network(CNN). CNN plays a significant role in improving the performance of object detection and recognition; CNN-based image recognition techniques have beaten human benchmark errors [20]. This notable viability has drawn attentions from other research areas, one of them being human detection and pose estimation [21–26].

DeepPose [21] is one of the pioneers for applying CNN to human pose estimation. It directly regressed the 2D coordinates of joints of human body parts. The DeepPose configured engineered training data, which were composed of the joints arranged in a kinematic tree resembling the human body so that it could replace any kinematic human body models.

Newell et al. [22] introduced a stacked hourglass network based on successive pooling and upsampling. The stacked hourglass network attempted to capture information across all scales from local features like faces, hands, or legs to full body poses and finally combine them to pixel-wise output. The pixel-wise output was interpreted by a heatmap for each joint, such as the neck, left elbow, right knee, left ankle, etc. The final human pose is estimated by the maximum activation across each heatmap.

Cao et al. [23] illustrated the CNN-based human part detection and association technique to detect multiple people in a given scene. The illustrated technique consists of two frameworks; one is for detecting human parts and another is for associating detected human parts into a human body. The first framework trained a deep learning network to provide both a set of confidence maps and a set of 2D affinity vectors for each human part. The confidence map suggests the candidate keypoints for part

detection, and the 2D affinity vector, what is called the part affinity field, provides the part association between a pair of two candidate keypoints. The inference of the trained network only provides the candidate key points and the candidate part association expressed by the 2D vector field. For human pose estimation, additional optimization is required to connect candidate part association to make a full body for each person. They proposed a greedy relaxation of the multi-dimensional matching problem that is basically the NP-Hard[32]. They built a minimal number of edges for a tree skeleton of human poses and then simplified the NP-Hard matching problem into a set of bipartite matching subproblems. Even though the proposed technique showed significant improvement in human pose estimation, some false cases are reported from rare poses or appearances as shown in Figure 2.3, which shows one leg limb (blue color) and one arm limb (green color) in the left image because the upright standing pose could be the dominant pose in the training dataset. Without the appropriate training samples for rare poses, it would be difficult to obtain the correct part association with the proposed affinity fields.

He et al. [27] introduced an extension of fast R-CNN [28], which is called mask R-CNN, that provides segmentation masks for each ROI (region of interest) on top of the bounding box and classification outputs. The proposed pixel-to-pixel alignment of ROI position can improve the accuracy of the location of the object. They also evaluated the generality of Mask R-CNN by applying the model to human pose estimation. They adopted Mask R-CNN to predict multiple masks, one for each keypoint that represents a human body part, e.g., left wrist, right hand, etc., and each mask is a one-hot mask that has only a one-pixel foreground. However, they only evaluated the detection accuracy of keypoint; the evaluation did not illustrate how to associate the detected human parts into a meaningful expression, the whole human body.

As inspired by 3D human modeling for human motion capture system and computer graphics, a technique [29] of corresponding between 2D images and 3D human body surface models has been developed. This dense correspondence between im-

Fig. 2.3. Examples of failure cases reported in [23]; failure by rare poses or appearances (right image) and by missing or false part detection (left image)

age pixels and surface points, which is called 'Densepose-RCNN', is the enhanced architecture of combining DenseReg [30] with Mask-RCNN.

All CNN-based human pose estimation essentially includes off-line training stages. It implicitly requires well-balanced training datasets that can cover as many conditions as possible, and, in general, huge amounts of data are required for training. Additionally, owing to the nature of supervised learning, it demands a lot of human labor to generate a large annotated dataset [31].

The recently proposed human pose estimations based on CNN have focused on pixel-wise segmentation and classification. Given an input image, they resulted in activated regions, which are usually specified as 2D coordinates or heatmaps, that are likely associated with human parts. All pixels in the resulting regions are classified with confidence factors. These outputs are only able to show the detected human parts individually; furthermore for human detection, some domain knowledge — e.g., a topological graph (tree) to configure human body as in [21,23,26,27] — will be applied to aggregate the individually detected human parts into whole human bodies. An alternate way is to design a structure of training data that can accurately represent the whole human body. One exemplary structure of training data consists of the human

part label, e.g., write, ankle, shoulder, etc., and regions where the corresponding human body parts are located [21]. The proposed online CMIL model is relatively less dependent on human labor for preparing the training data because it will generate its training data from the current inputs. This means that the proposed model can evolve during tracking. The new training data from each input make it possible to incrementally evolve the CMIL model. CNN-based models, which inevitably need off-line training, hardly change their network parameters during tracking. However, the CMIL model takes time to be stabilized because it is trained by samples collected in successive frames.

The proposed online CMIL model has no specific kinematic human body models to associate detected human body parts. Articulated human movements cannot be wholly supported by normal kinematic human pose structures which have been required to detect full human bodies. For instance, OpenPose [23] implicitly allows the directional association between detected body parts, e.g., the knee point as the start point and the ankle point as the end point, where the direction of 2D affinity vector is always from knee to ankle; however, Figure 2.3 shows the failure of pose estimation due to directional constraints on body part association. The proposed online CMIL model is intrinsically free from this constraint because there are no kinematic human body model assumptions.

2.3 Multiple Instance Learning

The recent machine learning techniques have been gradually used for object tracking due to increasing computational power that enables real-time learning. Supervised learning is one of most popular method in machine learning for object tracking. In supervised learning, the decision boundaries or concepts are determined by training samples that usually consist of features and known corresponding class labels. The performance of supervised learning relies on how sets of training samples can appropriately represent a space in which the samples to be classified are located. This

means that training samples should have correctly their corresponding class labels because incorrect class labels assigned to training samples cause a substantial bias for determining the decision boundary or concepts. However, class label assignment is a time-consuming task, especially when there exists a large amount of training samples. With the considerable volume of training samples, it is difficult to assign correct class labels of some training samples when these training samples are located around a true decision boundary in the sample space. Therefore, this ambiguity of class labels in training samples should be considered in the training stage.

Multiple Instance Learning (MIL) provides a good learning technique that effectively handles ambiguity in the learning samples. The MIL is a variant of the supervised learning process where classification labels are associated with sets of samples rather than individual samples. MIL had originally been suggested for pharmaceutical research area for estimating drug activities [32]. It is important to effectively consider deformations of molecular structure for finding candidate molecules to match a target drug because the final molecule has at least one deformation of the molecule to be matched in the target drug. From the view of generating learning samples, the candidate molecule with a large number of its variable structures can be analogous to a large set of learning samples and the positive class label can be assigned to the candidate molecule rather than one of the variable structures. In other words, the class label can be set to the bag of learning samples rather than each learning sample.

Given training samples $(\mathrm{x}_i, \mathrm{y}_i)$, the goal is to estimate a function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X} \in \mathbb{R}$ is the feature space with $\mathrm{x}_i \in \mathcal{X}$ and $\mathrm{y}_i \in \mathcal{Y}$ is the class label assigned to each data x_i. Let us assume that we have only two labels $\mathrm{y}_i \in \mathcal{Y} = \{+1, -1\}$. In MIL, the training data is not an individual sample but a set of training samples, called a bag, where each training sample in the bag is called as an instance. A bag B_i consists of a set of instances $\mathrm{B}_i = \{\mathrm{x}_{ij}\}_{j=1}^{N_i}$ and the label y_i of the bag B_i can be assigned by

$$\mathrm{y}_i = \max\{\mathrm{y}_{i1}, \mathrm{y}_{i2}, ...\mathrm{y}_{iN_i}\}, \tag{2.1}$$

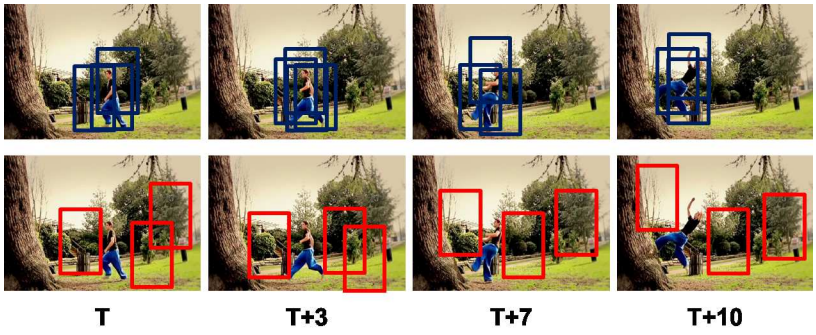


Fig. 2.4. Examples of positive bags and negative bags for tracking a man from T^{th} to $(T+10)^{th}$ frame. The first row shows the positive samples of the human to be tracked in blue rectangles, and the second row shows the negative samples of it in red rectangles. The positive bag B^{+}consists of positive samples that are patches shown in the blue rectangles in the first row.

where y_{ij} are the instance labels, which are unknown during the training stage. Each bag can be classified into two different bags: a positive bag and a negative bag. The positive bag $\mathrm{B}^{+} = \{\mathrm{x}_j\}_{j=1}^{N^{+}}$, i.e., the label $\mathrm{y} = +1$, has to contain at least one positive sample x_j whereas the negative bag $\mathrm{B}^{-} = \{\mathrm{x}_j\}_{j=1}^{N^{-}}$ consists of all negative samples. Figure 2.4 shows an example of both positive bags and negative bags in each frame.

We need to estimate the function $f : \mathcal{X} \rightarrow \mathcal{Y}$ given the training samples $\{(\mathrm{B}_i, \mathrm{y}_i)\}_{i=1}^{N}$. Numerous algorithms for solving MIL have been proposed [33–37] and Viola et. al [38] suggest, the MILBoost, which solves the MIL with training a boost classifier that maximizes the log likelihood of bags:

$$\log(\mathcal{L}(\mathrm{C})) = \sum_{i} \log(p(\mathrm{y}_i \mid \mathrm{B}_i)). \tag{2.2}$$

The likelihood of bag $p(\mathrm{y}_i \mid \mathrm{B}_i)$ can be expressed in terms of the likelihood of each instance. The Noisy-OR model is used for estimating the likelihood of the positive bag B_i,

$$p(\mathrm{y}_i = 1 \mid \mathrm{B}_i) = 1 - \prod_j (1 - p(\mathrm{y}_{ij} = 1 \mid \mathrm{x}_{ij})), \tag{2.3}$$

where y_{ij} is an output of classifier $\mathrm{y}_{ij} = \mathrm{C}(\mathrm{x}_{ij})$.

Under the Noisy-OR model, the likelihood of a set of training bags is

$$\mathcal{L}(\mathrm{C}) = \prod_i (p(\mathrm{y}_i \mid \mathrm{B}_i))^{\mathrm{y}_i} (1 - p(\mathrm{y}_i \mid \mathrm{B}_i))^{(1-\mathrm{y}_i)}, \tag{2.4}$$

where $\mathrm{y}_i \in \{0, +1\}$ is the binary label.

Using eq. (2.2), the log likelihood of the classifier is obtained by maximizing

$$\log(\mathcal{L}(\mathrm{C})) = \sum_i \mathrm{y}_i \log(p(\mathrm{y}_i \mid \mathrm{B}_i) + (1 - \mathrm{y}_i)(1 - p(\mathrm{y}_i \mid \mathrm{B}_i)). \tag{2.5}$$

The goal of the MIL process is to obtain in either instance classifiers or sample classifiers through maximizing the log likelihood of classifiers as described in eq. (2.5). Most MIL applications need to find out the instance classifiers for handling ambiguous class labels attached in the bag. Therefore, the class label of each instance can be determined during the training process.

In recent years, MIL combined with existing classification processes have addressed problems in object detection and tracking [34–36, 38]. Also, MIL can be implemented in online detection and tracking systems [37, 39], which allow us to update the reference model for tracking objects in frame by frame. We will focus on the problem of tracking location and scale of the bounding box surrounding humans of interest in the online video sequence. We assume that the initial position and size of the bounding box are given as prior knowledge for online learning. The online MIL can update the reference model of the human of interest in each frame which will be used for tracking the human in next frame.

2.4 Online Boosting

This section explains one type of ensemble learning algorithms: the boosting. We especially focus on describing the Adaptive boosting (Adaboost) learning algorithm. The Adaboost learning algorithm is a powerful learning technique, but it needs all training samples for building a classifier. Therefore, the online Adaboost learning algorithm is introduced for handling the case in which all training samples cannot be allowed while training the classifier.

2.4.1 Adaptive Boost

Ensemble learning is a combination of several primitive models whose individual outputs of classification or regression are aggregated for computing a final decision. The fundamental assumption is that each base decision model in the ensemble learning is uncorrelated with the others. However, this assumption can hardly be achieved because real learning data are generally located on a very complex feature space where each dimension cannot be easily distinguished. Therefore, many algorithms have attempted to generate as uncorrelated of base models as possible so that they guarantee the diversity of base models. The most popular ensemble learning algorithms are bagging and boosting: bagging (Bootstrap Aggregating) creates multiple sets of training samples with replacement and uses each of these sets to generate a classifier; boosting utilizes all training samples at each learning stage with different weight distribution over the training samples.

The boosting learning approach as an ensemble learning has shed significant light on the machine learning research since the late 1990s; the combination of weak classifiers provides a very powerful classifier under the shadow of a supervised learning framework. Freund et al. [40] have proposed an adaptive boost (AdaBoost) learning technique for improving the learning rate. Basically, the AdaBoost runs in offline mode, which means that all training samples should be available in the training stage. In each training iteration, each training sample is weighted by the currently estimated

weak classifier that will be added into a strong classifier which is composed of a collection of weak classifiers up to that moment. The details of the Adaboost algorithm are described in Alg. 1. Inputs of Adaboost are composed of three components: training sample with known labels ($\{(x_i, y_i\}_{i=1}^{N}$), a base learning model algorithm (i.e., the weak classifier, F_b) , and the number of base models to be combined as a final decision model (i.e., the strong classifier, M). The original Adaboost had been designed for a two-class problem, and later expanded the two-class Adaboost algorithm into a multi-class learning algorithm. The two-class Adaboost algorithm will be described in this thesis for its simplicity. The weighted distribution is set to be equally uniform for all training samples at an initial stage as $\frac{1}{N}$. At the first stage $t = 1$, the base model is trained by all training samples and the current weighted distribution. Next, the classification error of the current base model is computed as shown in Alg. 1 described in ϵ. The classification error is a sum of weights that correspond to the misclassified training samples in the current base model. If the classification error from the current base model is higher than $\frac{1}{2}$, the learning loop stops. Note that each base model in boosting requires that the classification performance needs to be better than random decision, which means the classification error has to be lower than $\frac{1}{2}$. If the classification error of the current base model is lower than $\frac{1}{2}$, the weight of the current base model is set to be $\frac{1}{2}\ln\frac{1-\epsilon}{\epsilon}$. After assigning weight to the current base model, the weighted distribution of all training samples is modified by the current base model performance. If the training sample can be classified correctly, the weight of this training sample will be smaller than the previous weight by multiplying an exponential margin. The weight of incorrectly classified training samples will be increased because these samples can probably be selected for training the next base model. The key factor of Adaboost is to assign the weight for each training sample in every iteration: the higher weights will be assigned to the training samples that the current base model fails to correctly classify. The falsely classified training samples are more likely to be selected for learning the base classifier at each iteration.

The training error bound of Adaboost is given by

Algorithm 1 Adaboost

Given inputs $\{(x_i, y_i)\}_{i=1}^{N}$, F_b , and M,

Initialize $D_1(n) = \frac{1}{N}$ for all $n \in \{1, 2, ..., N\}$.

For $t = 1, 2, ..., M$:

$h_t = F_b(\{(x_i, y_i\}_{i=1}^{N}, D_t)$

$\epsilon_t = \sum_{n:h_t(x_n \neq y_n)} D_t(n)$

if $\epsilon \geq \frac{1}{2}$ then,

$M = m - 1$ and terminate learning

else

$\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon}{\epsilon}$

$$D_{t+1}(n) = \begin{cases} \frac{1}{Z_t} D_t(n)\, e^{-\alpha_t y_t(n) h_t(x_n)} & \text{if } h_t(x_n) = y_n \\ D_t(n) & \text{otherwise} \end{cases}$$

end for

$H(x) = \text{sign}\left(\sum_{\text{t=1}}^{\text{M}} \alpha_{\text{t}} \text{h}_{\text{t}}(\text{x})\right)$

$$\widehat{error}(H) \leq \prod_t 2\sqrt{\epsilon_t(1-\epsilon_t)}, \tag{2.6}$$

where $\widehat{error}(H)$ is the training error of H.

As shown in Eq.2.6, Adaboost is not theoretically guaranteed the minimum training error but just the upper training error bound.

In addition, the upper bound of generalization error is

$$error(H) \leq \widehat{error}(H) + \mathcal{O}(\sqrt{\frac{Td}{m}}), \tag{2.7}$$

where $error(H)$ is the generalization error of the strong classifier H, T is the iteration number, d is the complexity of weak classifiers, and m is the number of training samples.

2.4.2 Online Adaboost

As explained in previous Section 2.4.1, Adaboost was originally used for offline learning algorithms because entire training samples should be required for determining the weights of each training sample and the strong classifier. The weight of each training sample assigns its probability as selected training samples for the current weak classifier that is added into the strong classifier.

Oza [41] first introduced the online Adaboost that does not require the entire set of training samples for learning weak classifiers. The online Adaboost has a fixed number of weak classifiers that will be trained from input samples in each iteration. Under the condition of using a naive Bayes classifier as the base classifier, the final classifier H_{online} of online Adaboost converges to that $H_{offiline}$ of the offline Adaboost as increasing the number of training samples, $N \rightarrow \infty$. The pseudocode of the online Adaboost is outlined in Alg. 2. The online Adaboost fundamentally works in a pre-defined set of base classifiers for each training sample. Each base classifier h_m in the pre-defined set has two types of weights: λ_m^{sc} and λ_m^{sw}. λ_m^{sc} is a sum of weight when the base classifier $h_m$correctly classifies the training samples and λ_m^{sw} is a sum

of weight when $h_m$incorrectly classifies them. Given the first training sample, the first base classifier $h_1$is learned by this training sample. If the $h_1$correctly classifies the given training sample, λ_1^{sc} is updated and the weight of this training sample λ is also updated from $\epsilon_1 = \frac{\lambda_1^{sw}}{\lambda_1^{sw}+\lambda_1^{sc}}$. If the $h_1$incorrectly classifies the given training sample, λ_1^{sc} is updated and the λ is also updated from $\epsilon_1 = \frac{\lambda_1^{sc}}{\lambda_1^{sw}+\lambda_1^{sc}}$. The given training sample and updated weight λ are passed to the next base classifier h_2 and then update the weight of base classifier h_2 as well as the weight of training sample λ. This process is conducted on all base classifiers in the pre-defined set, and then the final classifier is obtained by a sum of function of classification error in each base classifier, $h(x) = \text{argmax} \sum_{m:h_m(x)=y} \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$.

Grabner and Bischof [39] proposed a variant of online boosting for feature selection. They introduced an additional concept of training base classifiers: a selector. Given a pool of base classifiers $\mathcal{F} = \{h_1, h_2, ..., h_P\}$, a subset of base classifiers, $\mathcal{H}_k = \{h_1, h_2, ..., h_M\}$, is assigned to the selector S_k. The role of the selector is to pick up an exact one base classifier with minimum classification error to a given one training sample. Given a single training sample, the subset of base classifiers in the selector S_1 is trained and updates the importance weights of each base classifier. The updated importance weights of base classifiers are used for computing the classification error of each base classifier. The selector S_1selects the base classifiers with the minimum classification error and updates the importance weight of the current training sample. The above process iteratively continues in the selector $S_2, S_3, ..., S_N$. Note that the base classifier with the maximum classification error is replaced with new base classifier from the pool of base classifiers $\mathcal{F}$. The final classifier is obtained by linear combination of selectors,

$$H(x) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n h_n(x)\right). \tag{2.8}$$

The pseudocode of this online adaboost are outlined in Alg. 3. Compared to Oza's approach, they claim that the selection of the base classifier can be directly applicable to feature selection because each base classifier corresponds to one specific feature. It

Algorithm 2 Online Adaboost algorithm [41]

Initial $\lambda_m^{sc} = \lambda_m^{sw} = 0$ for all $m \in \{1, 2, ..., M\}$

Set the training sample's weight $\lambda = 1$.

For each base classifier h_m, $m \in \{1, 2, ..., M\}$in a base classifier pool $\mathbf{h}$

set k according to Poisson(λ_d)

Do k times

$h_m = F_o(h_m, (x, y))$

if $y = h_m(x)$

$\lambda_m^{sc} \leftarrow \lambda_m^{sc} + \lambda$

$\epsilon_m \leftarrow \frac{\lambda_m^{sw}}{\lambda_m^{sc} + \lambda_m^{sw}}$

$\lambda \leftarrow \lambda \left(\frac{1}{2(1-\epsilon_m)} \right)$

else

$\lambda_m^{sw} \leftarrow \lambda_m^{sw} + \lambda$

$\epsilon_m \leftarrow \frac{\lambda_m^{sw}}{\lambda_m^{sc} + \lambda_m^{sw}}$

$\lambda \leftarrow \lambda \left(\frac{1}{2\epsilon_m} \right)$

$h(x) = \text{argmax} \sum_{m:h_m(x)=y} \log \left(\frac{1-\epsilon_m}{\epsilon_m} \right)$

means that the final classifier can be the combination of features that are representative of the given classification space.

2.5 Tracking via Multiple Instance Learning

As described in Section 2.3, Babenko et al. [37] developed a robust tracking through an online updated appearance model sustained by Multiple Instance Learning, which we call Multiple Instance Learning Tracking (MILTracking). Basically, MILTracking is a variant of the Tracking-by-Detection approach; the reference template, i.e., the target model, is sequentially updated and subsequently used for finding the most similar appearance model in each frame. Given the updated target model, MILTracking finds the local image patch that has the most similar appearance. MILTracking uses a greedy motion model for discovering the best estimated position of the tracker, as shown in Fig. 2.5. The best estimated position of the tracker in time t is used for an initial position of the tracker in time $t + 1$. Within a search area, shown in a yellow dashed circle in Fig. 2.5, MILTracking cropped out a set of local image patches and then applied a strong classifier learned from previous time steps. Based on outputs of the strong classifier, MILTracking selects a local image patch that has the minimum distance from the target model in the appearance space, i.e., the most similar appearance. After locating the best position of the tracker in time $t + 1$, the MILTracking collected both positive samples and negative samples around the best position of the tracker, as shown in Fig. 2.5. These positive and negative samples are fed into the learning process in the MILTracking for updating the target model, in other words, the updated strong classifier. The MILtracking algorithm is summarized in Alg. 4.

In the training stage, MILTracking uses the online Adaboost that is modified from the proposed by [39]. The basic process is to select K weak classifiers from a pool of M classifiers in each step. As mentioned in [41], online Adaboost is a sub-optimal

Algorithm 3 Online Adaboost for feature selection

Initialize the importance weight $\lambda = 1$

for $n = 1, 2, ..., N$

  for $m = 1, 2, ..., M$

    $h_{n,m}^{weak} = F_o(h_{n,m}^{weak}, (x, y), \lambda)$

    if $h_{n,m}^{weak}(x) = y$

     $\lambda_{n,m}^{c} = \lambda_{n,m}^{c} + \lambda$

    else

     $\lambda_{n,m}^{w} = \lambda_{n,m}^{w} + \lambda$

    $e_{n,m} = \frac{\lambda_{n,m}^{w}}{\lambda_{n,m}^{w} + \lambda_{n,m}^{c}}$

  end for

end for

$m^{+} = \text{agrmin}_{\text{m}}(e_{n,m})$

$e_n = e_{n,m}$

$h_n^{sel} = h_{n,m^{+}}^{weak}$

if $e_n = 0$ or $e_n > \frac{1}{2}$ **then** exit

$\alpha_n = \frac{1}{2}\ln(\frac{1-e_n}{e_n})$

if $h_n^{sel}(x) = y$

  $\lambda = \lambda\frac{1}{2(1-e_n)}$

else

  $\lambda = \lambda\frac{1}{2e_n}$

end if

$m^{-} = \text{argmax}_{\text{m}}(e_{n,m})$

$\lambda_{n,m^{-}}^{c} = 1, \lambda_{n,m^{-}}^{w} = 1$

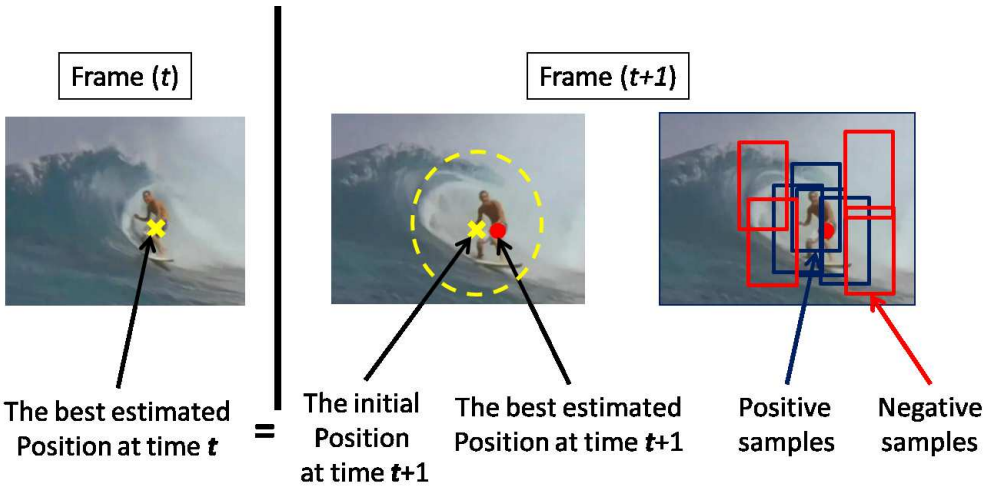


Fig. 2.5. Tracking with the greedy motion model

Algorithm 4 MILTracking algorithm

INPUT: a video frame at time t and a final strong classifier $\mathbf{H}_{t-1}$

OUTPUT: a final strong classifier $\mathbf{H}_t$ and the best position of object $\ell(\mathbf{x}_t^*)$

1. Extract a set of local image patches, $\mathrm{X} = \left\{\mathbf{x} \mid \left\|\ell(\mathbf{x}) - \ell(\mathbf{x}_{t-1}^*)\right\| < d\right\}$
2. Compute the feature vector of each image patch in the set X
3. Apply the classifier $\mathbf{H}_{t-1}$ for estimating $p(y = 1 \mid \mathbf{x})$ for $\mathbf{x} \in \mathrm{X}$
4. Find the best position of object, $\ell(\mathbf{x}_t^*) = \arg\max_{\ell(\mathbf{x})} p(y = 1 \mid \mathbf{x}) \mid \mathbf{x} \in \mathrm{X}$
5. Collect positive samples, $\mathrm{X}^p = \{\mathbf{x} \mid \|\ell(\mathbf{x}) - \ell(\mathbf{x}_t^*)\| < s\}$ and negative samples, $\mathrm{X}^{\mathrm{neg}} = \{\mathbf{x} \mid s < \|\ell(\mathbf{x}) - \ell(\mathbf{x}_t^*)\| < t\}$
6. Update the final strong classifier $\mathbf{H}_t$

procedure because all training samples are unavailable at specific times; only a partial training sample is available at each time step.

3. COMPONENT-BASED MULTIPLE INSTANCE LEARNING

The work described in this chapter deals with the problem of tracking human movements when the articulated motions span large variations in the configuration space. Our approach is based on discriminative learning called Multiple Instance Learning (MIL) [32, 33] that associates class labels with sets of instances (as opposed to single instances in the traditional learning algorithms). We first define the new basic learning unit of MIL tracking for resolving tracking failures when the large articulated human movements exist. We then apply this new MIL tracking for various articulated human movement tracking and show how it improves performance.

3.1 Introduction

In recent years, MIL combined with existing classification strategies has been used to address problems in object detection and tracking [34, 35, 38]. Work has also been carried out in using MIL for online detection and tracking [37, 39]. In this work, the reference model used in the tracking process is dynamically updated from frame to frame. Specific focus of our work described in this chapter is on addressing the problem of tracking human movements that cause large frame-to-frame variations with the following steps: (1) Project the most probable bounding box in the previous frame into a candidate bounding box in the current frame; (2) Apply a strong classifier computed in the previous frame to different bounding boxes in the vicinity of the candidate box and select the most probable bounding box; (3) Construct randomly displaced versions of this most probable bounding box as positive examples; (4) Choose image patches from the current frame outside the region spanned by the positive examples as negative examples; (5) Apply an image segmentation algorithm

to the positive and the negative examples; (6) Use the component based boosted MIL algorithm to assign probabilities to the different positive examples with regard to the similarity of their pixel distribution to the most probable bounding box in the current frame; and, finally, (7) Compute the strong classifier from the component based boosted MIL algorithm.

With the conventional logic of object tracking described above, we introduce the new basic learning unit of MIL tracking. In our approach, image patches in the positive and the negative examples are subject to automatic segmentation to yield what we call components. It is the components that are subject to the basic classification by the boosted MIL algorithm. The components give our approach extra degrees of freedom that are not possessed by the previous use for boosted MIL for object tracking. For articulated motions, these additional degrees of freedom can play a critical role in transferring the probabilities over the positive examples in the previous frame to those in the current frame. When articulated motions are involved, it is less likely that pixel brightness distributions would match well between two corresponding bounding boxes in two successive frames. However, if we first segment each of the bounding boxes into components based on, say, approximate uniformity of brightness levels, we are more likely to find component-to-component matches between the two corresponding bounding boxes. This then underlies the rationale behind our component based boosted MIL algorithm. From the next section, we first provide a brief introduction to MIL and then show how this technique can be adapted for tracking humans with large articulated motions.

3.2 Multiple Instance Learning

As described in Section 2.3, MIL is a discriminative learning algorithm that is more accommodating of ambiguities in the training data than the conventional approaches. Given training samples (x_i, y_i), the goal is to estimate a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X} \in \mathbb{R}^d$ is the feature space with $x_i \in \mathcal{X}$, and $y_i \in \mathcal{Y}$ is the class label assigned to each

data x_i. In MIL, the training data is not an individual sample but a set of the training samples called a bag and each training sample in the bag is called an instance. A bag X_i consists of a set of instances $X_i = \{x_{ij}\}_{j=1}^{N_i}$ and the label y_i of the bag X_i as assigned by $y_i = \max_j(y_{ij})$, where y_{ij} is the instance label, which is unknown during training stage. Each bag is either a positive bag or a negative bag. The positive bag $X^+ = \{x_j\}_{j=1}^{N^+}$ has to contain at least one positive sample x_j whereas the negative bag $X^- = \{x_j\}_{j=1}^{N^-}$ consists of all negative samples.

We need to estimate the function $f : \mathcal{X} \to \mathcal{Y}$ given the training samples $\{(X_i, y_i)\}_{i=1}^{N}$. Numerous algorithms for solving the MIL problem have been proposed [35–37] over the years. Of particular interest to us is the solution proposed by Viola et al. [38]. Their algorithm, called MILBoost, solves the MIL problem by training a boosted classifier that maximizes the log likelihood of the bags:

$$\log(\mathcal{L}(C)) = \sum_i \log(p(y_i \mid X_i)), \tag{3.1}$$

where y_{ij} is an output of classifier $y_{ij} = C(x_{ij})$. The likelihood of the bag $p(y_i \mid X_i)$ can be expressed in terms of the likelihood of each instance. The likelihood of a set of training bags is given by

$$\mathcal{L}(C) = \prod_i (p(y_i \mid X_i))^{y_i} (1 - p(y_i \mid X_i))^{(1-y_i)}, \tag{3.2}$$

where $y_i \in \{0, 1\}$ is a binary label. The likelihood $p(y_i \mid X_i)$ represents the connection with the bag X_i and its instances x_{ij} in terms of noisy-OR model

$$p(y_i = 1 \mid X_i) = 1 - \prod_{j=1}^{N_i} (1 - p(y_{ij} = 1 \mid x_{ij})). \tag{3.3}$$

3.3 Component Based Online Boosted MIL Tracking

However, it is significantly required to update the reference template in appearance based tracking. The update of reference template under machine learning framework needs to run learning stages in each iteration in online mode: the learning samples

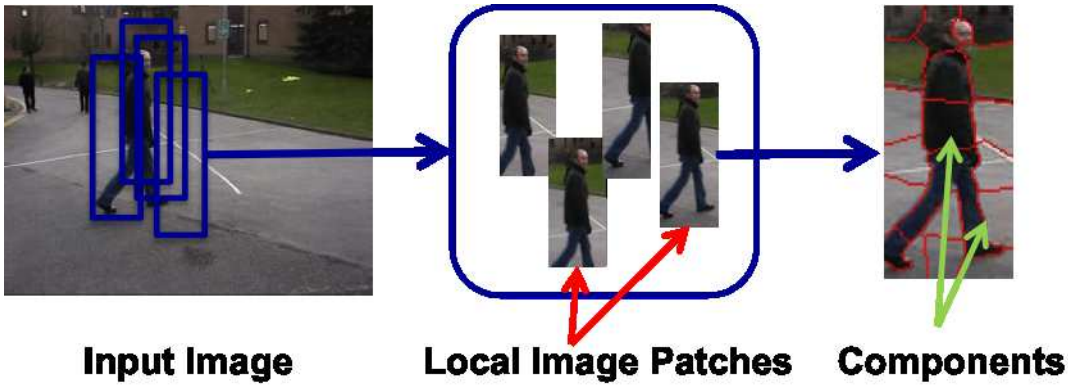


Fig. 3.1. Multiple image patches are first extracted from the input image. Automatic segmentation is then applied to each image patch to yield discriminative clusters, or components.

should be collected in each frame sequentially. Also, the online learning framework inevitably has a partial learning samples. Therefore, we need some online learning framework to handle partial collections of learning samples that are collected sequentially.

A straightforward application of MILBoost fails to track human movements when large articulated motions are involved on account of excessively large variations in the appearance that needs to be tracked. To augment the logic of MILBoost in order to more explicitly take into account the articulation caused variations in the images, we now introduce a component based approach to the boosting of MIL in which the components provide us with the additional degrees of freedom to accommodate the image variations caused by articulations. By components we mean the blobs produced by an image segmentation algorithm when applied to the patch inside a bounding box. In this manner, our algorithm can simultaneously address the sort of appearance variations handled by MILBoost and the human articulations that we need for tracking people movements.

To explain how the components for the different image patches are used, in each frame we start with the extraction of a number of image patches based on the current bounding box for the human being tracked. As in the regular MIL boosting, some of these patches are considered to be positive examples and the others negative examples. Each patch is segmented into components, as shown in Figure 3.1. While the appearance variations are addressed by collections of local image patches (i.e., positive instances), the human articulations are implicitly dealt with through the components. We can think of the components as implicitly representing the human parts. In this manner, our proposed boosted MIL approach can use the components in both the positive and the negative examples for accommodating human articulation changes during tracking.

To give the reader a sense of motion articulations as captured by the image components extracted from a patch, we show Figure 3.2. If we simply represent the human figure by a single patch containing a silhouette, as shown in Figure 3.2(b), the fore-

ground/background segmentation shown in the figure that is used as a silhouette for tracking usually contains insufficient information for tracking the movements. Even a parts based approach may fail to track in such cases on account of the "broken linkages" shown in Figure 3.2(c) when we fit an articulated model to the silhouette. However, the components in the image patch, as shown in Figure 3.2(d), can implicitly represent human articulation with loosely-connected configurations in that each component contains partial or whole human parts.

We will now extend the MIL formalism of the previous section in order to incorporate the components in it. We now consider a bag X_i to consist of a set of sub-bags: $X_i = \{X_{ij}\}_{j=1}^{N_i}$, and a sub-bag X_{ij} a set of component instances: $X_{ij} = \{x_{ijk}\}_{k=1}^{N_{ij}}$. The Noisy-OR model of MIL in Eq. (3.3) can be expanded into

$$p(y_i \mid X_i) = 1 - \prod_j (1 - p(y_{ij} \mid X_{ij})), \tag{3.4}$$

$$p(y_{ij} \mid X_{ij}) = 1 - \prod_k (1 - p(y_{ijk} \mid x_{ijk})). \tag{3.5}$$

The likelihood of the positive bag X_i for the component based boosted MIL is expressed by

$$\begin{aligned} p(y_i \mid X_i) &= 1 - \prod_j (1 - (1 - \prod_k (1 - p(y_{ijk} \mid x_{ijk})))) \\ &= 1 - \prod_j (\prod_k (1 - p(y_{ijk} \mid x_{ijk}))) \end{aligned} . \tag{3.6}$$

The likelihood of each component can be represented by the logistic regression model:

$$p(y_{ij} = 1 \mid x_{ij}) = \sigma(H(x_{ij})) = \frac{1}{1 + e^{(-H(x_{ij}))}}. \tag{3.7}$$

Under the tracking-by-detection paradigm for component based boosted MIL tracker, we need to update the classifier in every frame with selected positive and negative bags. In every frame we detect the position of image patch at which the output of the proposed boosted MIL classifier, learned in the previous frame, is maximized. The optimal detected position $\hat{p}$ of image patch is computed by

$$\hat{p} = \underset{p \in S}{\operatorname{argmax}}\, p(y \mid X^p), \tag{3.8}$$

where S is a predefined search area, X^p is the image patch at location p, and $p(y \mid X^p)$ is the output of proposed boosted MIL classifier. For collecting learning samples, given the estimated optimal position $\hat{p}$, we collect positive instances (image patches) whose center positions are located in S; negative bags are generated from image patches whose center positions are located outside of S. Figure 3.3 shows an example of collecting positive instances around the human and negative instances from the background area.

As shown in Algorithm 1, we extend the MILBoost [37] into component based boosted MIL. Given a pool of K candidate weak classifiers h, the algorithm chooses M weak classifiers $\mathbf{h}$ from the candidate pool by optimizing a specific objective function J,

$$\mathbf{h}_k = \underset{h \in \{h_i\}_{i=1}^{K}}{\operatorname{argmax}} J(\mathbf{H}_{k-1} + h), \tag{3.9}$$

where $\mathbf{H}_{k-1}$ is the strong classifier up to the first $(k-1)$ weak classifiers. The M weak classifiers are selected by noisy-OR model of probability of component labeling rather than those of image patches.

Algorithm 5 Component Based Online Boosted MIL

Input: Given N training samples (X_i, y_i) where image patches X_i and their correspondent label $y_i \in \{0, 1\}$,

1: Apply superpixel algorithm to obtain components $\{x_{ij}\}_{j=1}^{N_i}$ in i^{th} image patch X_i and set label y_{ij}, $\forall j, y_{ij} = y_i$

2: Extract features $\{f_{ijk}\}_{k=1}^{K}$ from each x_{ij}

3: Update all K weak classifiers in the pool with all data $\{x_{ij}, y_{ij}\}$

4: For each weak classifier,

5: Compute likelihood of components in each image patch

6: Compute likelihood of image patch by combining likelihood of components

7: Choose a classifier having maximum likelihood

8: Select the classifier and add to a strong classifier

9: Repeat until collecting M classifiers.

Output: $H(x) = \Sigma h_k(x)$

3.4 Experimental Validation

We present the result of CMIL tracking on four video sequences that include the large articulated human movements, namely (1) ETH sunny day sequence (TUD)[1], (2) UIUC1 standing to sit sequence($StandToSit$)[2], (3) Skating sequence from VTD webpage($Skating$)[3], and (4) Gymnastic athlete sequence from Youtube(Gym)[4].

We compare our proposed algorithm with MILBoost tracking algorithm [37]. The initial position of human in each sequence is manually specified. Each weak classifier h_k is composed of the log odd ratio,

$$h_k(x) = \log\Big[\frac{p(y=1 \mid x)}{p(y=0 \mid x)}\Big]. \tag{3.10}$$

We assume that both $p(y=1 \mid x)$ and $p(y=0 \mid x)$ are of a normal distribution, $N(\mu_1, \sigma_1)$ and $N(\mu_0, \sigma_0)$, respectively. Those parameters are updated in each frame by gathering positive and negative components.

$$\begin{aligned} \mu &\leftarrow \gamma\mu + (1-\gamma)\frac{1}{n}\sum f_k(x), \\ \sigma &\leftarrow \gamma\sigma + (1-\gamma)\sqrt{\frac{1}{n}\sum (f_k(x)-\mu)^2}, \end{aligned} \tag{3.11}$$

where $0 < \gamma < 1$ is a learning rate parameter.

We used the turbopixel algorithm [42] for extracting components from each image patch. Haar-like features are extracted in each component. We represent each component as a vector of Harr-like feature, which are randomly generated, similarly to [18]. Each feature consists of two to six rectangles, and each rectangle has a real valued weight. The feature value is then a weighed sum of the pixels in all the rectangles. These features can be computed efficiently using the integral image trick described in [38]. The parameters were set as follows: the search radius s is set to 36 pixels. We

[1]http://www.vision.ee.ethz.ch/~aess/dataset
[2]http://vision.cs.uiuc.edu/projects/activity
[3]http://cs.snu.ac.kr/research/~vtd
[4]http://www.youtube.com/watch?v=FJIrHgshB20

sample positive in each frame using a radius r=4. This generates maximum 50 image patches as the positive bags. The 65 image patches are randomly sampled as negative bags. The learning rate γ for the weak classifiers is set to 0.85. Finally, the number of candidate weak classifiers M was set to 50 and the number of chosen weak classifiers K was set to 10. We compared the tracking results in two different categories: a qualitative comparison and a quantitative comparison. Fig. 3.4 shows the qualitative results of tracking humans in four video sequences. The blue line shows the tracking result from the CMIL tracker and the yellow line shows those from the MILBoost tracker. Both algorithms successfully track the walking man in the street in the first test sequence. However, in other three test sequences, the proposed CMIL tracking algorithm shows the better tracking performances than MILBoost tracking. Especially the MILBoost tracking lost the human just after a few frames from the initial frame in the Skating sequence and Gymnastic sequence that have large articulated human movement. Table 3.1 shows the quantitative results of two compared trackers. The quantitative tracking results are measured by averaging pixel distance between the ground truth position and the output of each tracker through whole frames. The ground truth position of human in the sequences are manually extracted. As shown in the qualitative comparison, the quantitative comparison clearly indicates that the CMIL tracking achieves better performance for tracking humans with large articulated movements: the pixel distance errors of CMIL tracking are a half time lower than those of MILBoost tracking.

Fig. 3.5 and Fig. 3.6 display the selected positive components in the estimated bounding box enclosed the human to be tracked. Even the MIL tracker lost the tracking target because of partial occlusion during tracking, the proposed CMIL tracker could successfully track the target. This may rely on the fact that each component can be described by a set of features and corresponding classifiers which could memorize a sort of unique information for corresponding components.

Table 3.1.

A quantitative evaluation by comparing mean pixel error. The mean pixel error was computed by measuring the distance between the center position of the bounding box and that of the human being tracked which is manually picked.

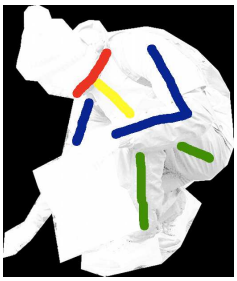
Video Sequence		TUD	StandToSit	Skating	Gym
Mean Pixel Error	MILBoost	13.7	9.57	25.68	35.65
	Component based boosted MIL	9.1	9.46	10.86	19.26

3.5 Conclusion

In this chapter, we presented a component based version of the boosted MIL algorithm for tracking articulated human movements. Compared to the conventional boosted MIL tracking algorithm, our proposed method can better handle large articulated motions that cause significant variations in the appearance of the humans that need to be tracked. Components in each image patch that are automatically generated by image segmentation provide the additional degrees of freedom that the tracker needs to deal with the large frame-to-frame variations caused by articulated movements. We validated the advantages of our proposed method by comparing the tracking results against the MILBoost algorithm on four different video sequences.

(a)

(b)

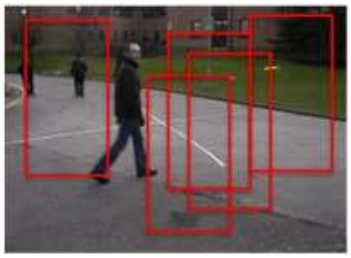


(c)

(d)

Fig. 3.2. Estimation of articulation of human pose. (a) an input image patch; (b) an obscure silhouette from foreground/background segmentation; (c) broken linkages and self occlusion in a possible joint model representation; (d) components in the image patch.

(a)



(b)

Fig. 3.3. Examples of candidate positive instances depicted by blue rectangles in (a) and candidate negative instances depicted by red rectangles in (b).

Fig. 3.4. Comparison of tracking results by MILBoost (yellow line) and the proposed component based boosted MIL (blue line). From the first row, *TUD*, *StandToSit*, *Skating*, and *Gym* sequence are used for comparing results.

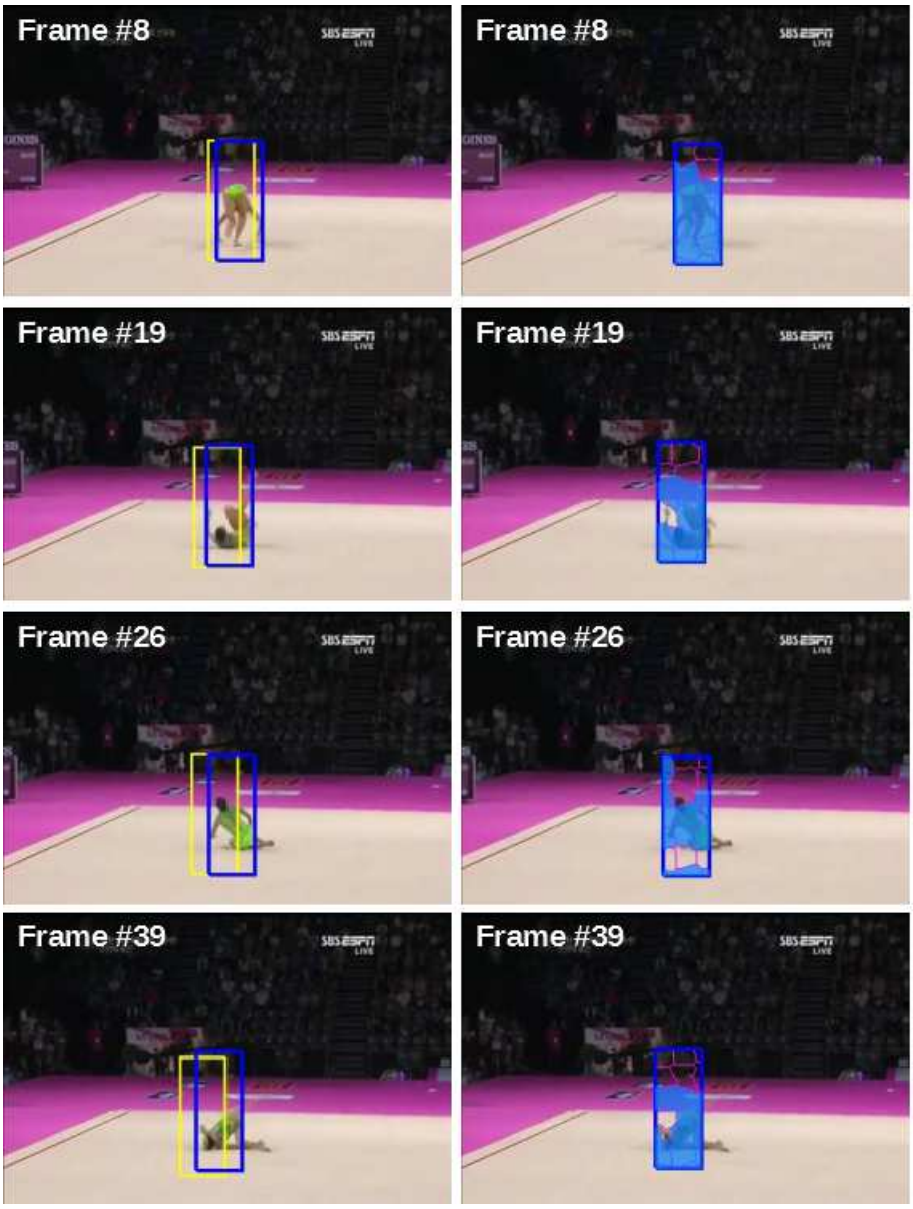


Fig. 3.5. Selected positive components (blue mask) in the estimated bounding box.

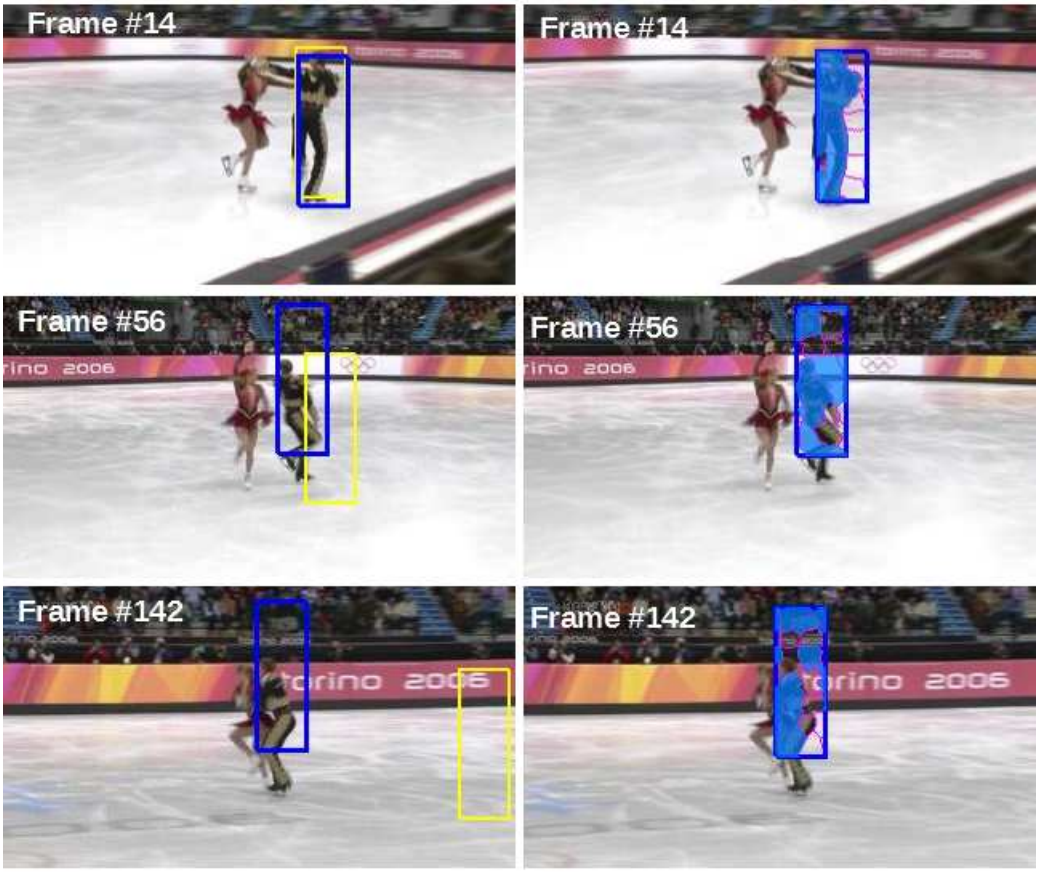


Fig. 3.6. Selected positive components (blue mask) in the estimated bounding box.

4. POSITIVE BAG CONFIGURED COMPONENT-BASED MIL

This chapter primarily focuses on developing an extended component MIL tracking with alternative positive image patches. The alternative positive image patch is defined by the specific number of positive components rather than at least one positive component. This specific number of positive components, which can declare the positive image patch, can be adaptively varied during tracking in accordance with positive and negative component distributions in the given image patch. We showed how to define the alternative positive image patch with respect to positive component distribution and the improving tracking performance.

4.1 Introduction

Tracking humans is an important issue in several computer vision applications, such as surveillance, activity recognition, human behavior analysis, etc. Despite some successes, the performance of the current breed of human tracking algorithms leaves much to be desired, especially when one factors in illumination changes, background clutter, large articulated motions, etc. [1].

A typical approach to tracking humans depends on an appearance model as represented by a reference template that is created in the first frame of a video sequence. This reference template is basically an intensity distribution within an area surrounding the human being tracked, the area specified by a bounding box. The goal of the tracker is to track this reference intensity distribution over the successive frames in the video sequence. However, when the human being tracked engages in large articulated motions, the intensity distribution as defined by the reference template is inadequate for describing the distributions corresponding to all the motions of the

articulated parts. Motions such as bending, crunching, kneeing, etc., produce large changes in the overall shape of the human as seen by the camera. Our work focuses on how to handle such large appearance variations common to articulated human motions in order to improve tracking performance.

In order to deal with the problems of precisely labeling the positive and negative images in several computer vision tasks, such as image retrieval, text recognition, object tracking, etc., researchers [37, 39] have suggested the use of a discriminative learning approach called Multiple Instance Learning (MIL) [32,33] in which the classifier is trained from positive and negative bags of instances (as opposed to the positive and the negative instances directly in the more traditional approaches). Since a bag is a set of instances and since the positive bags are allowed to contain both positive and negative instances (whereas the negative bags must only contain negative instances), MIL is more forgiving of the errors made in choosing the positive instances for training. As demonstrated in Chapter 3, the fact that MIL is more accommodating of the errors in identifying the image parts that may be on the object being tracked allows it to be used for the tracking of relatively large articulated movements by humans. More specifically, the contribution in Chapter 3 used a version of MIL known as online boosted MIL for demonstrating the human tracking results. The boosted MIL was used in a component-based framework (CMIL), with the components being the result of automatic image segmentation applied to the current frame of the video sequence. The classification output for components as determined by MIL was used to update the reference template for the human being tracked.

In investigating the failure modes of the algorithm, we have noticed its performance suffers from selecting positive samples from the image background region due to overly cluttered backgrounds, too large frame-to-frame illumination changes, or improper sampling range for positive samples. Figure 4.1 presents two results obtained with CMIL as an example of the failure mode caused by a cluttered image background. They are basically the same articulated movement by the human subject. The results shown in the right column are not as robust as those shown in the

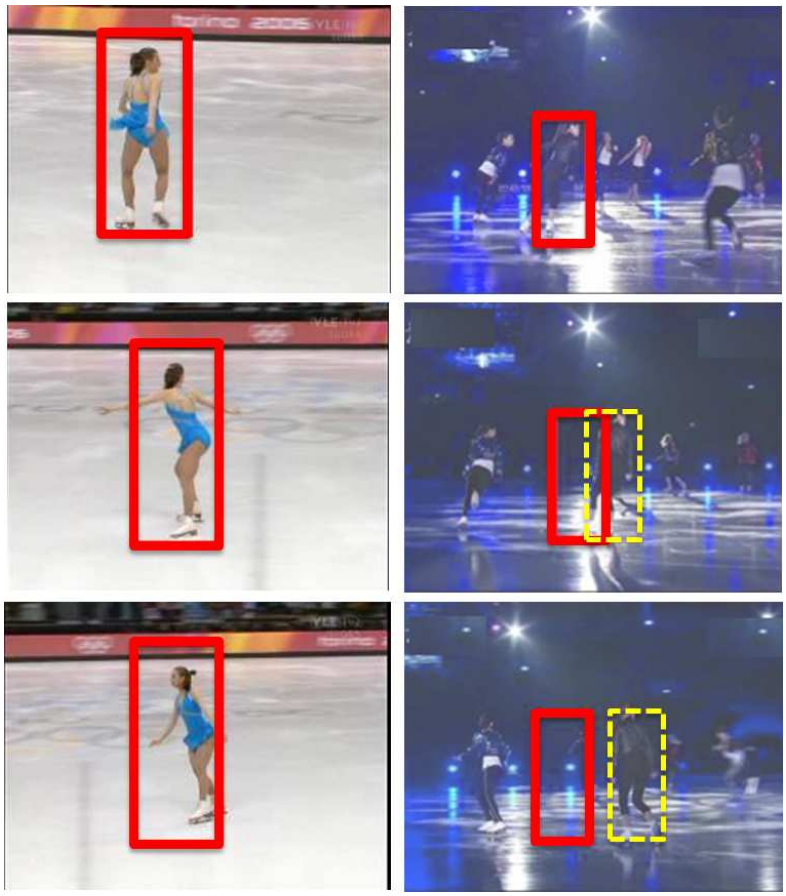


Fig. 4.1. An example of the failure modes of CMIL tracking for articulated human motions. CMIL shows the different tracking results on the same articulated movement. The left column displays the precise tracking results, and the right column depicts the failure of tracking results. The red solid rectangles represent the tracking results of CMIL in both columns. The yellow dashed rectangles in the right column represent the ground-truth.

left column on account of the indistinguishability of the colors and textures on the human vis-a-vis those in the background.

Our investigation shows that this failure mode of the CMIL can be attributed directly to one of the foundational assumptions of MIL: that for a bag to be positive it needs to contain only one positive instance at the least. This rather minimalist requirement creates many false positives, meaning many components that are labeled

as belonging to the object as opposed to belonging to the background, in the tracking process. Figure 4.2 shows an example of a false positive image patch with only a couple of positive components which represent a fraction of the human body parts. These examples clearly provide evidence that it needs a new configuration of positive bags.

Recently, Li and Vasconcelos [43] presented the soft bag configuration for MIL, in which the positive soft bag is defined as the bag with more than μ number of positive instances, which they called the μ-positive bag. They demonstrated that this soft bag performs better compared with the conventional MIL for scene classification. However, all experiments have been conducted offline with all possible training samples, and the optimal μ values are determined by empirically varying μ values from 1 to 20. The goal of our present work is to demonstrate the use of MIL for tracking articulated motions with a relaxation of the foundational assumption mentioned above. We introduce the notion of a minimum positive count as the minimum number of positive instances that must exist in a bag for it to be called positive. In contrast to the soft bag with μ-positive bag, we show how this new parameter, the minimum positive count, can be determined adaptively on a frame-to-frame basis depending on the conditions prevailing in the frame. We make this parameter a function of the ratio of the total number of positive and negative components declared in the previous frame.

4.2 CMIL tracking with adaptively configured positive bags

In this section, we first describe the notion of a minimum positive count for adaptive component-based MIL (aCMIL). Note that we can call it also as component-based MIL with adaptive positive count (CMIL-AC). Then, we discuss how it is computed dynamically in each frame and used for adaptively configuring positive bags in a MIL-based tracking framework.

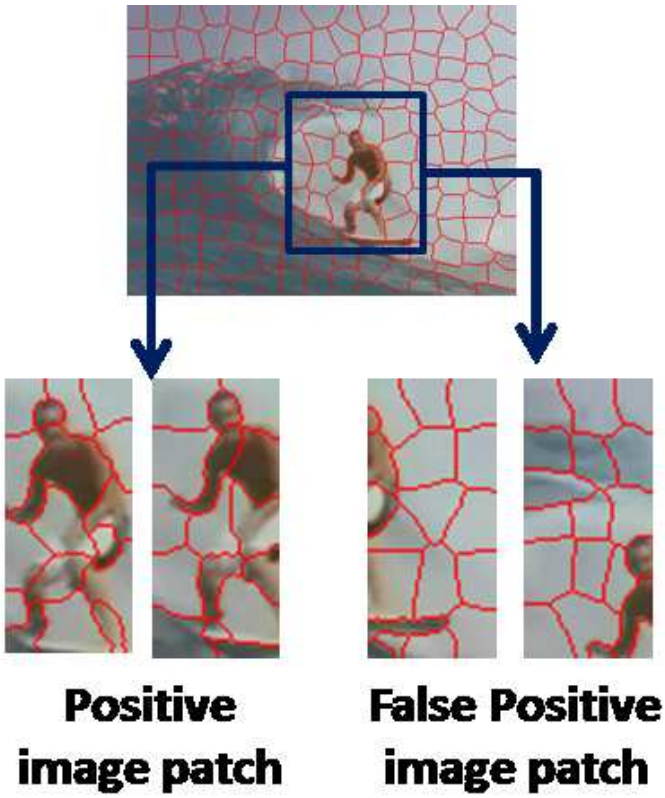


Fig. 4.2. An example of a false positive image patch under the foundational assumption of MIL. The image patch with only a couple of positive components (a fraction of the human body parts) can be classified as a positive image patch.

4.2.1 Minimum Positive Count of a Positive Bag

Given the local image patch $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ and its label $y \in \{-1, +1\}$, we need to assign the proper class label of the image patch $\mathbf{x}$ by taking into account the class label y_i of each component x_i. From a probabilistic viewpoint, the likelihood probability of the positive image patch, $\Pr(y \mid \mathbf{x})$, is conventionally represented by the Noisy-OR model [33],

$$\Pr(y \mid \mathbf{x}) = 1 - \prod_{i=1}^{n}(1 - \Pr(y_i \mid x_i)). \tag{4.1}$$

The Noisy-OR model represents the foundational assumption of MIL: the local image patch is positive if there exists at least one positive component in the image patch. However, the Noisy-OR model for MIL is not suitable for CMIL tracking when samll number of positive components exist in the positive image patch. It is more natural to assign the positive label to the image patch that has at least a specified number of positive components. From the above observations, the proposed assumption of aCMIL is that the positive image patch has to contain at least the minimum number of positive component, which we call the minimum positive count (τ). The minimum positive count can provide a flexible way of assigning a positive label to image patches to the extent of the distribution of both positive and negative components in image patches. In addition, the minimum positive count helps aCMIL build a more acceptable decision boundary, as depicted in Fig. 4.3. We need to adaptively change the minimum positive count in each frame during tracking to account for varying distributions of positive and negative components. The details of how to change the minimum positive count is explained in 4.2.2.

Under the new assumptions of aCMIL, the likelihood of a positive image patch can be computed as follows. Each component in the image patch has its corresponding confidence score (z_i) which is a real-valued output of a classifier. Since we do not know the true label of each component in the training stage, we estimate the label of the component by looking at the sign of its confidence score from the current classifier.

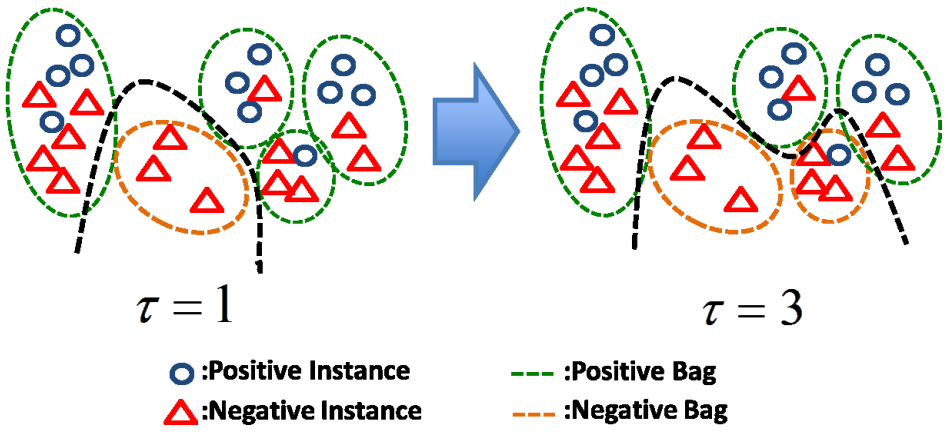


Fig. 4.3. The minimum positive count affects the decision boundary (black dash lines) of the two-class classification. As the minimum positive count τ changes from one to three, the aCMIL algorithm results in a much better shape of the decision boundary.

Given that the set of confidence scores in an image patch is $\mathbf{z} = \{z_1, z_2, \ldots, z_n\}$, a positive component count constraint in the image patch, $\mathrm{N_p}$, is computed as

$$\mathrm{N_p} = \sum_{i=1}^{n} (c(z_i)) - \tau, \qquad c(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{4.2}$$

where τ is the minimum positive count. Therefore, the image patch is labeled as positive when $\mathrm{N_p}$ is positive. In other words, the image patch is labeled as positive when there exist at least τ positive components. Along with the positive component count constraint ($\mathrm{N_p}$), the most likely positive component [44] can be another contribution to the likelihood of a positive image patch as

$$\Pr(y = 1 \mid \mathbf{z}) \propto \mathrm{e}^{z_{max}}, \tag{4.3}$$

where $z_{max} = \max\{z_1, z_2, \ldots, z_n\}$.

Based on the positive component count constraint ($\mathrm{N_p}$) and the most likely positive component (z_{max}), the likelihood of the positive image patch is expressed by a logistic function,

$$\Pr(y \mid \mathbf{x}) = \frac{1}{1 + \exp(g(\mathrm{N_p}, z_{max}))}, \tag{4.4}$$

where $g(\mathrm{N_p}, z_{max})$ is a function of the new assumptions of aCMIL. In this paper, we define the function $g(\mathrm{N_p}, z_{max})$ as

$$g(\mathrm{N_p}, z_{max}) = -\mathrm{N}_p \times z_{max}. \tag{4.5}$$

4.2.2 Adaptive Configuration of Positive Bags

For the aCMIL under the proposed assumption described in Section 4.2.1, each positive image patch has at least the minimum positive count of positive components. Therefore, the minimum positive count plays a critical role in collecting positive image patches. A fixed minimum positive count can generate false positive image patches

because distributions of positive and negative components in the image patch vary in each frame due to the variations of the shape and appearance of the human being tracked. As a result, the minimum positive count should be adaptively changed in order to prevent generating false positive image patches. The adaptive changes of the minimum positive count results in a more precise decision boundary through the reduction of false positive image patches, as depicted in Fig. 4.3. In this paper, the distributions are simply estimated by sample mean and variance.

The minimum positive count needs to be increased in order to avoid generating false positive image patches if the negative components are dominant in the positive image patches. On the other hand, the minimum positive count needs to be decreased in order to prevent the problem of degeneracy, which produces too few positive image patches when the positive components are dominant. We can derive a new minimum positive count based on both measured variance and mean of label of components in the i^{th} patch,

$$\tau_i = \frac{n_i}{\lambda}(1 + \frac{\sigma_i^2}{\mu_i^2}), \tag{4.6}$$

where λ is a user-defined parameter, n_i is the total number of components in the i^{th} patch, and μ_i and σ_i^2 are the mean and the variance of the label of components, respectively.Note that theoriginal label $y \in \{-1, +1\}$ is converted into the label $y' \in \{0, 1\}$

4.2.3 Online aCMILBoost

We extend the online MILBoost [37] by constructing an alternative likelihood model with adaptively configured positive bags. A pseudo code of our proposed method is shown in Algorithm 6. Given a pool of K candidate weak classifiers h, the algorithm chooses M weak classifiers $\mathbf{H}$ from the candidate pool by optimizing a specific objective function J, for example a concave function,

$$\mathbf{H}_k = \underset{h \in \{h_i\}_{i=1}^{K}}{\operatorname{argmax}} \left\{ J(\mathbf{H}_{k-1} + h) \right\}, \tag{4.7}$$

where $\mathbf{H}_{k-1}$ is the strong classifier up to the first $(k-1)$ weak classifiers. The M weak classifiers are selected by the sum of the log-likelihood of each image patch; each log-likelihood is computed by the positive component count constraint($\mathrm{N_p}$) and the confidence response of the most likely positive component (z_{max}) as described in Eq. (4.4). At each i^{th} positive image patch, the minimum positive count (τ_i) is updated by the mean and variance of components in the image patch. Then we estimate a final minimum positive count (τ) through the weighted sum of the minimum positive count in each image patch.

Both positive and negative patches are collected through a simple motion model for tracking. In each frame, we find the image patch that most likely encapsulates the human being tracked by determining the output of classifiers trained in the previous frame. The optimal position of the image patch can be formulated by

$$\hat{p} = \underset{p \in \mathcal{S}}{\operatorname{argmax}} \left\{ \Pr(y \mid \mathbf{x}^{(p)}) \right\}, \tag{4.8}$$

where $\hat{p}$ is the optimal position of the image patch, $\mathcal{S}$ is a predefined search area, $\mathbf{x}^{(p)}$ is the image patch at location p, and $\Pr(y \mid \mathbf{x}^{(p)})$ is the output of the aCMILBoost classifier. After finding the optimal position of the image patch enclosing the human being tracked, both positive and negative learning samples are generated around the neighborhood of the optimal position of the image patch.

4.3 Experiments

We tested our proposed tracker, the adaptive component-based MIL tracker (aCMILTrack) on some challenging video sequences, namely *Gym*[1], *Skating*[2] , *StandToSit*[3] , and *Wiggle*[4] sequences. These sequences include extensive articulated human movement such as bending, rolling, handstand, etc. We compare our algorithm to the

[1]http://www.youtube.com/watch?v=FJIrHgshB20
[2]http://cs.snu.ac.kr/research/~vtd
[3]http://vision.cs.uiuc.edu/projects/activity
[4]http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/

Algorithm 6 Online aCMILBoost with adaptively configured positive bags

Input: Data set $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where i^{th} patch $\mathbf{x}_i = \{x_{i1}, x_{i2}, ...\}$, $y_i \in \{0, 1\}$ and τ

Update all M weak classifiers in the pool with data $\{x_{ij}, y_i\}$

Initialize $\mathbf{H} = 0$

for $k = 1$ to K **do**

  for $m = 1$ to M **do**

  $z_{ij}^m = \mathbf{H} + h_m(x_{ij})$

  $z_{\max}^m = \max\{z_{i1}^m, z_{i2}^m, ..., z_{ij}^m\}$,
  $\mathrm{N_p} = \left(\sum_{j=1}^{n} \frac{\mathrm{sgn}(z_{ij}^m)+1}{2}\right) - \tau$

  $p_i^m = \sigma(\mathrm{N_p} \times z_{max}^m)$

  $L^m = \sum_i (y_i \log(p_i^m) + (1 - y_i)\log(1 - p_i^m)$

  end for

$m^* = \arg\max_m \{L^m\}$

$h_k(x) \leftarrow h_{m*}(x)$

$\mathbf{H} \leftarrow \mathbf{H} + h_k(x)$

end for

for $i = 1$ to N **do**

  $\tau_i = \frac{n_i}{\lambda}(1 + \frac{\sigma_i^2}{\mu_i^2})$

end for

$\tau = \sum_i \alpha_i \tau_i$

Output: Classifier $\mathbf{H}(x) = \sum_k h_k(x)$ and updated τ

online MIL tracker (MILTrack) [37] and component-based MIL tracker (CMILTrack) in Chapter 3.

For a fair comparison, all parameters of the learning classifiers in all the trackers are fixed in all the test sequences except for the parameter λ in aCMILTrack, described in Eq. (4.6). The qualitative evaluation is accomplished by displaying the bounding box around the best estimated position of the human being tracked with articulated motions in each frame as shown in Fig. 4.4, Fig. 4.5 , Fig. 4.6 , and Fig. 4.7. The quantitative evaluations are achieved by two different measures: average location error and precision. The average location error is computed by averaging the pixel distance between the center of the bounding box that most likely surrounds the human being tracked and that of the ground truth bounding box, which are manually collected in a whole video sequence. However, the average location error is not enough to show the performance of the tracker. We additionally include precision plots, similar to the ones described in [37]. The precision plots show the percentage of frames in which the distance between the center position of the estimated bounding box and the ground truth position was within some threshold distance. Furthermore, the precision is calculated from frames in which the detected human was located in the bounding box because the error distance is valid only when the detected human is inside the bounding box.

Figure 4.8 shows the location error in each frame in four different test sequences. Table 4.1 summarizes the average location errors. aCMILTrack shows the lower location error in the Gym, Skating, and Wiggle test sequences. In the StandToSit sequence, aCMILTrack and CMILTrack have very similar location errors. In the precision plots shown in Fig. 4.10, aCMILTrack shows better precision performance at a lower threshold distance, which indicates that the estimated bounding box satisfies the small location error as well as the precise encapsulation of the human being tracked. In the Wiggle sequence, both the average location error and the precision of CMILTrack are worse than those of MILTrack. However, aCMILTrack shows significantly improved performance with respect to both location error and precision.

Table 4.1.
Average location error with respect to pixel distance in each test sequence.

	Gym	Skating	StandToSit	Wiggle
MILTrack	24.11	33.14	19.19	13.11
CMILTrack	19.76	27.94	13.62	19.6
aCMILTrack	15.97	16.45	13.32	5.83

It implies that the minimum positive count compensates for the foundational assumption of MIL in articulated human motion tracking. Note that aCMILTrack uses a fixed size of bounding box specified in the first frame. Figure 4.12 displays the adaptive changes of the minimum positive count with respect to the distribution of positive components, measured by the mean of positive components, in the Skating sequence. The minimum positive count varies with respect to the mean of the positive component in positive patches. The minimum positive count increases as the mean of the positive component decreases because aCMIL pursues greater confidence for estimated positive patches by increasing the minimum positive count, which is the threshold to classify positive and negative patches. In contrast, an increase in the mean of the positive components induces the decrease of the minimum positive count around 30-40 frames in Figure 4.12.

4.4 Conclusion

In this Chapter, we presented component-based MIL with adaptively configured positive bags for tracking extensive articulated human movements that cause large appearance variation of the human being tracked. The proposed approach allowed us to change the required minimum number of positive components in a positive sample in order to filter out false positive samples in both learning and classifying input samples in each frame. The required minimum number of positive components can

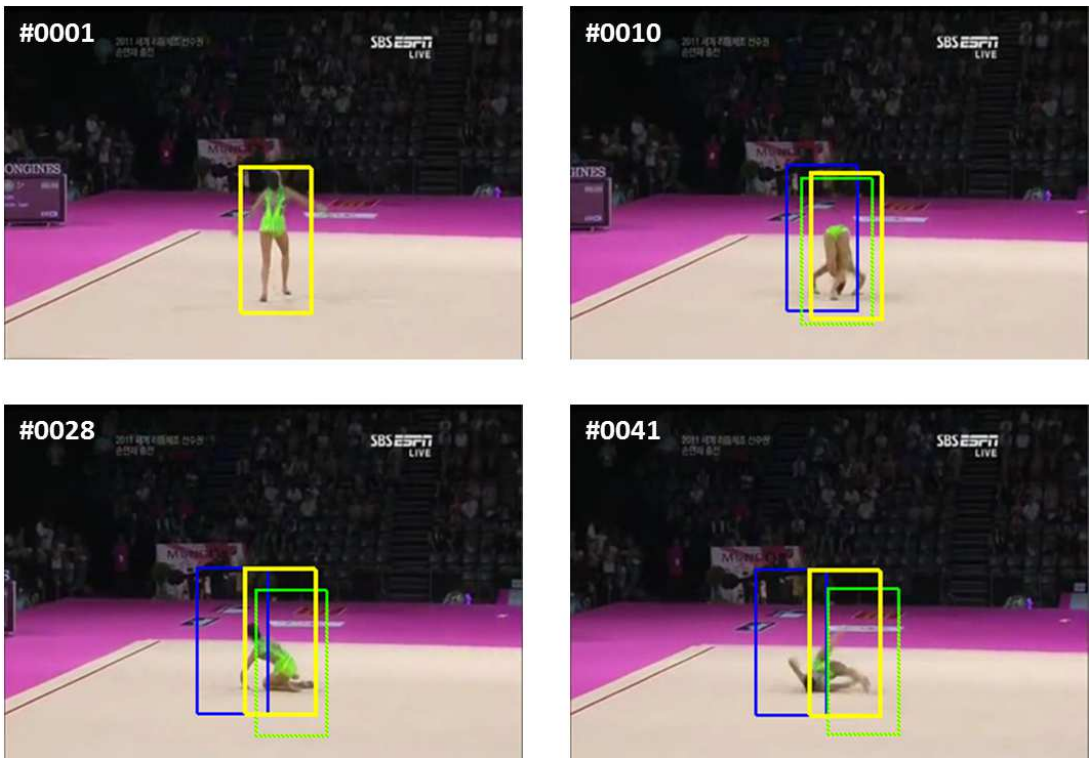


Fig. 4.4. Tracking results on the Gym test sequence. We compared three different trackers, MILTrack (Blue), CMILTrack (Green), and aCMILTrack (Yellow).

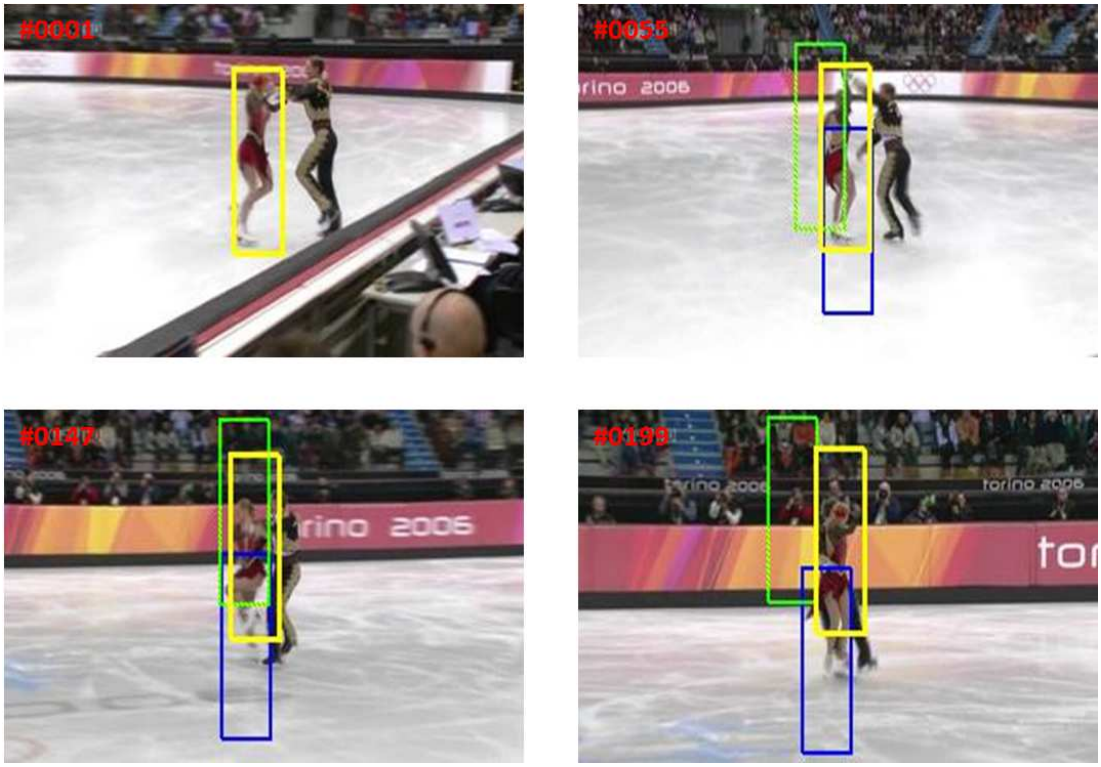


Fig. 4.5. Tracking results on the Skating test sequence. We compared three different trackers, MILTrack (Blue), CMILTrack (Green), and aCMILTrack (Yellow).

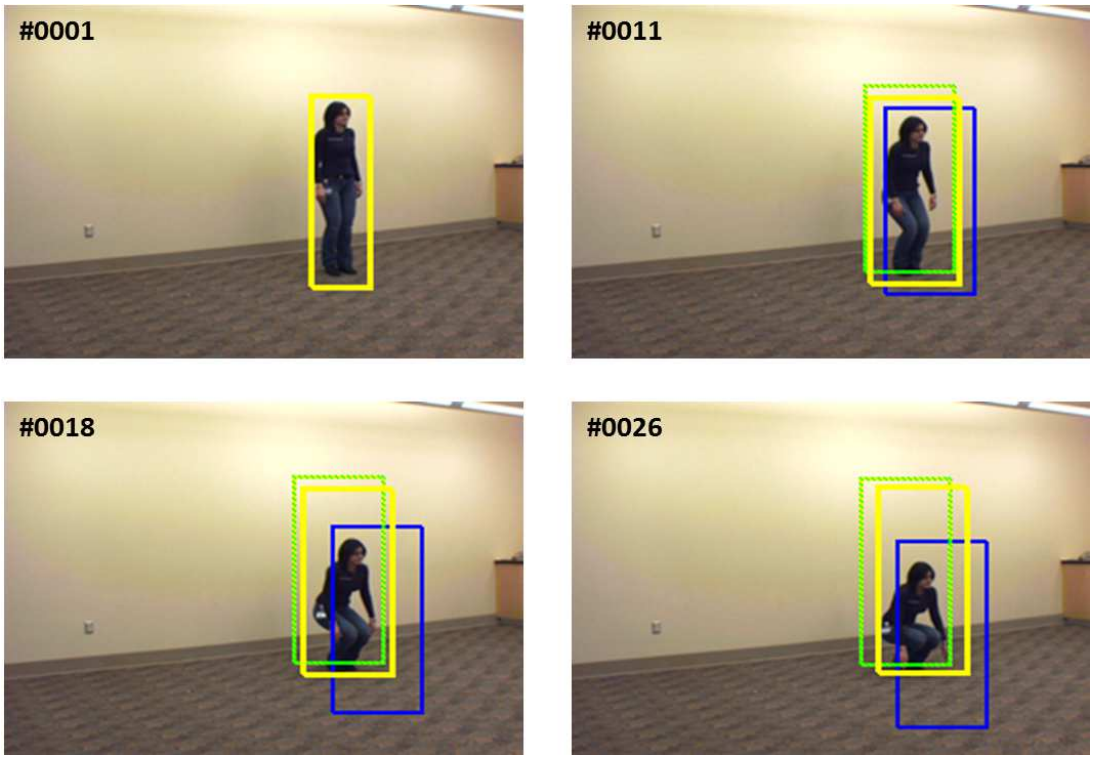


Fig. 4.6. Tracking results on the StandToSit test sequence. We compared three different trackers, MILTrack (Blue), CMILTrack (Green), and aCMILTrack (Yellow)

Fig. 4.7. Tracking results on the Caviar Wiggle test sequence. We compared three different trackers, MILTrack (Blue), CMILTrack (Green), and aCMILTrack (Yellow)

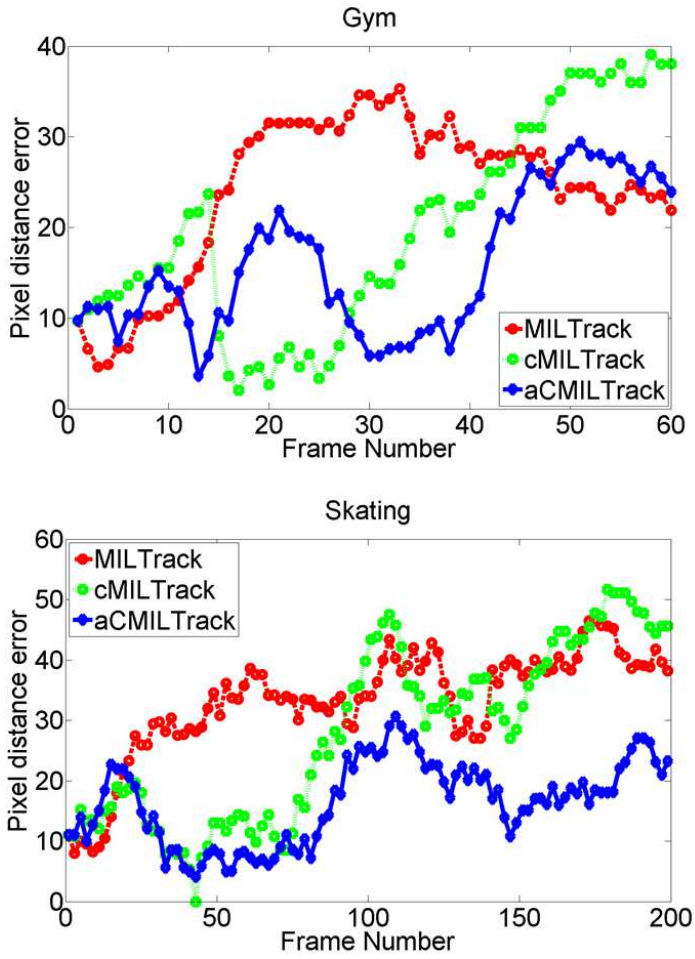


Fig. 4.8. Pixel distance error from different trackers on four test sequences.

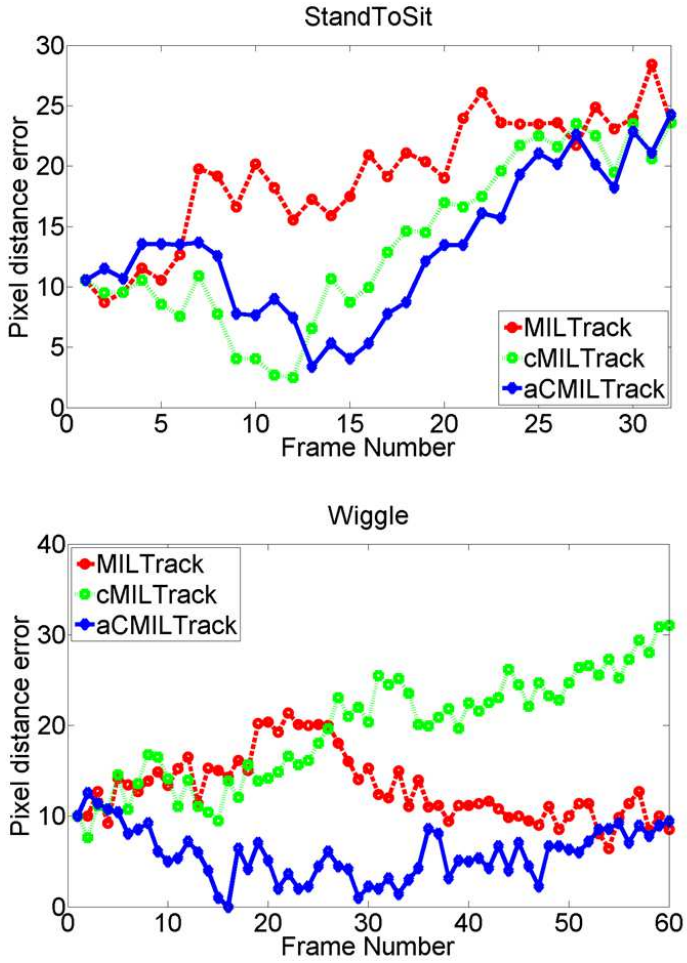


Fig. 4.9. Pixel distance error from different trackers on four test sequences.

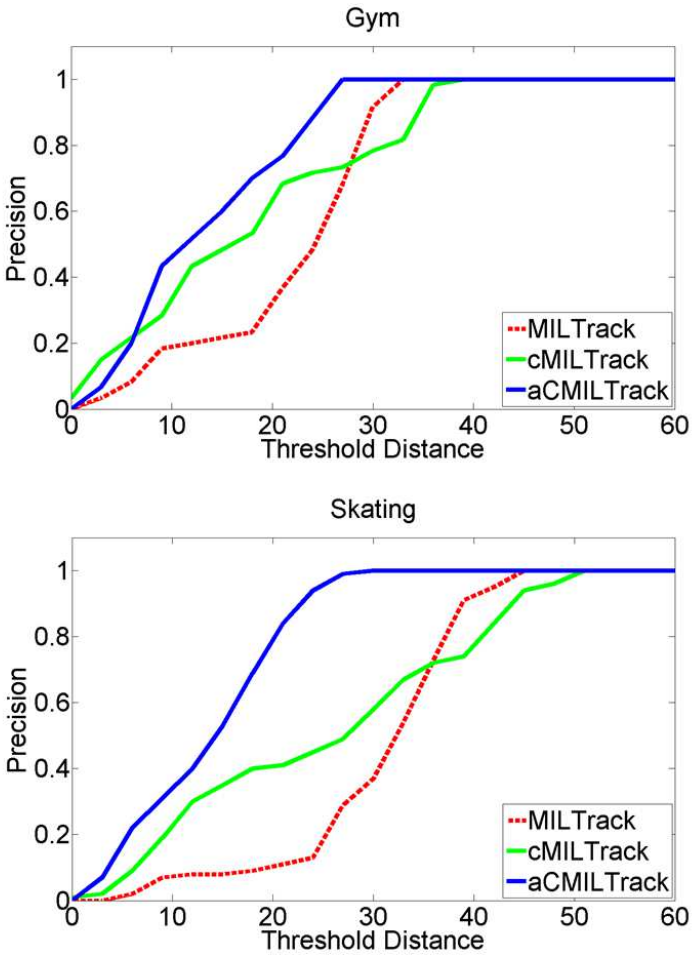


Fig. 4.10. Precision plots for validating different trackers.

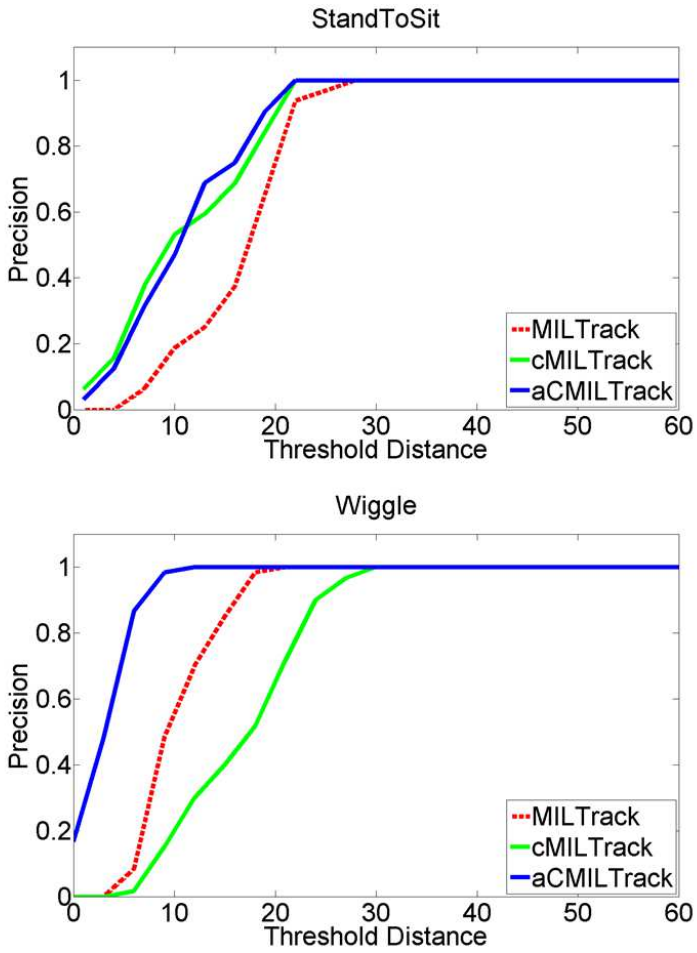


Fig. 4.11. Precision plots for validating different trackers.

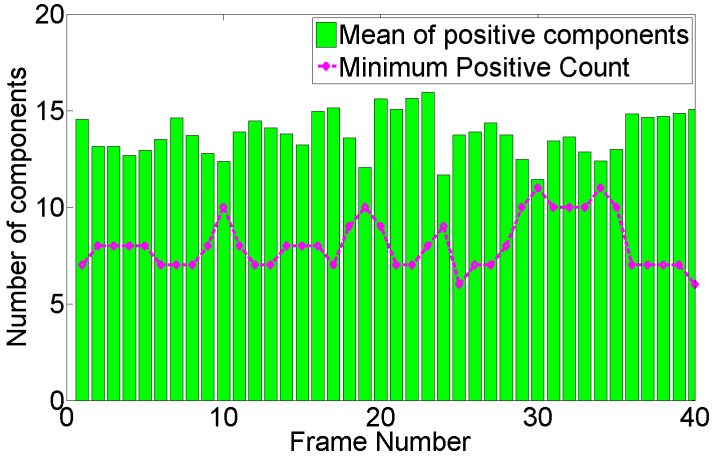


Fig. 4.12. An example of adaptive changes of the minimum positive count in the Skating sequence.

be determined by the distribution of positive and negative components in positive image patches. We evaluated our proposed tracker by comparing against other existing tracking algorithms in video sequences containing extensive articulated human movements. Our proposed tracker demonstrated better tracking performance with respect to both location error and precision.

5. COMPONENT-BASED MULTIPLE INSTANCE LEARNING WITH PARTICLE FILTERING

This chapter is primarily based on a published paper at WACV 2014 titled "Robust tracking of articulated human movements through Component-Based Multiple Instance Learning with particle filtering." [46] The main contribution in this chapter is a robust approach for tracking human subjects as their limbs and torso are engaged in large articulated movements while the entire body is executing a large translational motion with respect to the pointing angle of the camera. For large translational motions by the target, the CMIL tracking works with a motion prediction framework to more accurately estimate the most probable positions of the target in the next frame of a video sequence: this prediction is carried out with a particle filter. This coupling between the CMIL tracking and the particle filter yields more accurate estimates of candidate positions of the target in the next frame given the position of the target in the current frame.

5.1 Introduction

As discussed before, tracking humans that one sees most often in the literature usually involve only simple motions such as walking and running. Since these algorithms depend straightforwardly on projecting — in some cases after taking into account the motions estimated for the subject — a bounding box at the location of the subject in the current frame into a bounding box in the next frame, they fail for obvious reasons when the human subjects are executing large articulated motions [1].

There do exist tracking approaches that come under the label "Tracking by Detection" that can be expected to work more robustly in the presence of large articulated motions. In general, their performances depend on the accuracy with which

the various components of the articulated object being tracked can be detected in a frame under the typical constraints of real-time processing of a video stream. The accuracy versus time tradeoff in these algorithms can be improved by organizing the components in some manner in one frame for the purpose of searching for their correspondents in the next frame. Approaches based on Multiple Instance Learning (MIL) [32, 33] are one way to solve this problem of how to lend some organization to this frame-to-frame search for the different components of an articulated object.

The work described in Chapter 3 presents a variation on the basic MIL tracker to make it more suitable for tracking large in-place articulated motions, i.e., CMIL. This approach has a key step of automatic segmentation to the positive and negative instances. The segmentation of positive and negative instances yields components that lend themselves more easily to the delineation of the pixel blobs in the next frame. Whereas the CMIL approach in Chapter 3 works well for large but in-place articulated motions, it breaks down for obvious reasons when the human subject is engaged in large translational motions while his/her limbs and torso are engaged in large articulated movements.

Both the originally-proposed MIL based tracking [37] and the CMIL based tracking assume that the frame-to-frame variations in the center of mass of the target are sufficiently small so that the most probable bounding box in the next frame can be located simply by projecting the most probable bounding box in the current frame and searching in the vicinity of the projected bounding box. However, this assumption is violated if the human subject is also executing large translational motions with respect to the camera pointing angle.

Obviously then, when a human subject is executing large translational motions while engaged in articulated movements of his/her limbs, head, and torso, we must combine the CMIL algorithm with a motion prediction framework. The contribution is to demonstrate how to combine the CMIL based tracking with the motion prediction framework, a particle filter.

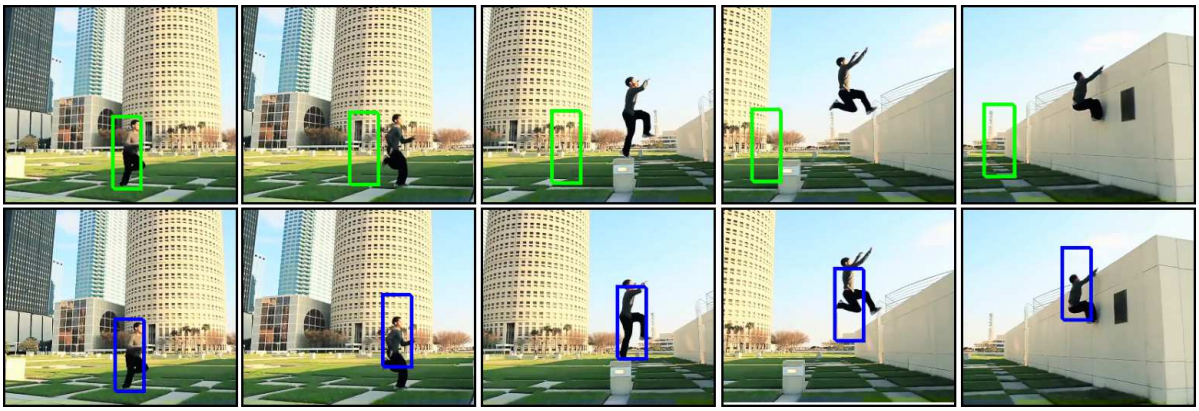


Fig. 5.1. Note that the human subject is engaged in both the articulated movements made by his entire body while he is also executing a large translational movement with respect to the pointing angle of the camera. The sequence of images in the top row is for the case when tracking is attempted with just the CMIL based tracker. And the sequence of images in the bottom row is for the case when we combine particle-filter based motion prediction with CMIL based tracking.

The top row of images in Figure 5.1 visually illustrates how easy it is to lose a track when a motion prediction framework is not used to augment a CMIL based tracker. The entire body of the human subject is undergoing large articulated movements while the center of the blob of the pixels occupied by him is moving rapidly with respect to the pointing angle of the camera. However, when we include a particle-filter based motion prediction in the tracking algorithm, the CMIL tracker produces the excellent tracking results shown in the bottom row of images in the same figure.

The remainder of this chapter is organized as follows: In Section 5.2, we review the previously related work. Section 5.3 presents brief overviews of the CMIL based tracking and a particle-filter based tracking. In Section 5.4, how CMIL can be combined with a particle-filter based prediction framework. We present quantitative and qualitative experimental results produced by this combination tracker in Section 5.5. Finally, we conclude in Section 5.6.

5.2 Related Works

Since the inspiration for the work reported in this Chapter has come from how various researchers have combined novel approaches to target modeling with the Bayesian logic of particle filtering, we provide a brief review of such literature here. Another major source of our inspiration was how researchers have combined the particle-filter based tracking with binary classification logic for more robust tracking. In the rest of this section, we first quickly review the former and mention two specific contributions in which the human body was represented by a set of blobs. Subsequently, we describe the previous work on combing particle filtering with binary classification logic that is more along the lines of our own contribution.

For human tracking, Lee et al. [47] have shown how a particle filter can be combined with a parts-based human tracker in which the human body is modeled as a set of parts and each part considered a particle in the particle filter. Along the same lines, Isard and MacCormick [48] have used a particle filter framework for multi-blob track-

ing with the blob likelihood representing the frame-to-frame location of the human body.

With regard to combining particle filtering with binary classification logic for more robust tracking, Okuma et al. [49] have demonstrated an AdaBoost based approach that is used to combine the results obtained by a mixture of particle filters. The proposal distribution in their work is represented by a linear combination of the prior transition distribution and a Gaussian distribution corresponding to the detections at the output of the AdaBoost algorithm. Along similar lines, Li et al. [50] have proposed a framework for low frame rate video tracking that is based on using AdaBoost to combine three distinctive classification algorithms, LDA, offline AdaBoost, and an online-learned AdaBoost, in a three-stage cascade implementation. On the other hand, the contribution by Song et al. [51], deals specifically with the dynamic nature of the changes to the particles and also the number of particles as a target is being tracked. The particles in this approach are selected on the basis of the weights assigned to them by an SVM classifier. As the target is being tracked, the particles that become invisible for one reason or another are assigned uniform weights. Closer to home, there is the work reported in [52, 53] in which an appearance based model of the target is learned and and continually updated with MIL and the model then tracked with a particle-filter based tracker. The MIL framework in these contributions helps the tracker cope with noisy nature of on-line learning.

In relation to the contributions mentioned above, our goal in this paper is to demonstrate that when we combine the CMIL based tracking with motion prediction made possible by particle filtering, we get a truly robust tracker that can deal with the body articulation by a human target as the target is moving across the field of view of a camera.

5.3 Brief Reviews of MIL, CMIL, and Particle Filtering for Tracking

5.3.1 MIL and CMIL Based Tracking

The main idea of MIL is to learn the best class labels for a set of data instances from what are known as the positive and the negative *bags* of such instances. A positive bag must contain at least one truly positive instance and all of the instances in a negative bag must be negative instances. The advantage of the MIL approach is that it is more accommodating of the errors made in labeling positive instances during the learning process. This gives an MIL-based object tracker, as, for example, originally formulated by Babenko et al. [37], the freedom to make errors when declaring certain blobs in the next frame as positive instances of the most probable blob in the current frame. All that is needed is that a positive bag for localizing the target in the next frame contain at least one truly positive instance. This can be ensured by straightforwardly projecting the most probable bounding box in the current frame into the next frame and creating several candidate bounding boxes by shifting this projected bounding box around. Assuming that the frame-to-frame motion of the target is small, we can be reasonably certain that the set of bounding boxes thus created in the next frame will contain at least one truly positive instance of the target. To ensure that all the instances in a negative bag are negative, all we have to do is to choose these instances relatively far from where the target being tracked is likely to be.

A fundamental step in the MIL-based tracker is the transfer of probabilities to one or more candidates for positive instances in the next frame from one or more positive positive instances in the current frame. In the past, this has been carried out on the basis of pixel brightness similarities between the bounding boxes. However, this logic for establishing similarities breaks down when a human subject is executing large articulated motions. In such cases, estimating similarity probabilities on the basis of the similarity of pixel brightness levels — especially when such calculations involve all of the pixels inside the bounding boxes — just does not work. In CMIL, this prob-

lem was taken care of by applying automatic segmentation to the bounding boxes in the current frame to yield what the authors of [45] refer to as the components. We have argued, since each segmented component is likely to have a fairly uniform color, establishing correspondences between the positive instances in the current frame and the next is likely to be more accurate than when the same calculations are carried out with all of the pixels inside the bounding boxes. This allowed us to demonstrate tracking results when the human subjects were executing large articulated motions. For the underlying MIL logic, the CMIL was implemented using the MILBoost algorithm of Viola et al. [38]. The label assigned assigned to a candidate positive instance in their implementation was a "noisy-or" of the labels of the components contained therein.

5.3.2 Particle Filter Based Tracking

For a quick review of the formulas that go into a prediction framework based on particle filtering, consider a sequence of the state vectors $\{\mathbf{x}_t \mid t \in \mathbb{N}\}$ and another sequence of the observation vectors $\{\mathbf{z}_t \mid t \in \mathbb{N}\}$ that we may associate with a target in motion. We assume that the time evolution of the state vectors is described by a possibly nonlinear function f_t as shown below (this being referred to as the *process model*):

$$\mathbf{X}_t = f_t(\mathbf{X}_{t-1}, \mathbf{V}_t), \tag{5.1}$$

We may also associate an observation model with the sequence of state vectors:

$$\mathbf{Z}_t = g_t(\mathbf{X}_t, \mathbf{W}_t), \tag{5.2}$$

where g_t is also possibly a nonlinear function. The time varying variables $\mathbf{V}_t$ and $\mathbf{W}_t$ are the white noise and the observation noise.

From a Bayesian perspective, the tracking problem is to recursively compute a Bayesian estimate for $\mathbf{x_t}$ given the observations $\mathbf{z}_{1:t}$ up to time t. The prediction problem is to construct a Bayesian estimate for $\mathbf{x_{t+1}}$ given the observations $\mathbf{z}_{1:t}$ up

to time t. Thus it is required to construct the posteriori state probabilities for either $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ or $p(\mathbf{x}_{t+1}|\mathbf{z}_{1:t})$ given all the causal observations $\mathbf{z}_{1:t} = (\mathbf{z}_1, \cdots, \mathbf{z}_t)$. With the Markov assumption on state transitions, it can be shown that the posteriori probability $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ can be expressed in the following form:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1}}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})}, \tag{5.3}$$

where $p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$ is the normalization constant.

In particle filter, in general, the posteriori probabilities for $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ are approximated by a set of N samples and their weights w, $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}_{i=1}^N$, as follows:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}). \tag{5.4}$$

For object tracking in videos, one commonly uses the sequential importance resampling (SIR) particle filter for removing what is known as the degeneracy problem. This entails giving equal weights to the particles. As a result, the equation shown above can be written as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} \frac{1}{N} \delta(\mathbf{x}_t - \hat{\mathbf{x}}_t^{(i)}). \tag{5.5}$$

In the prediction stage of an SIR particle filter, one starts by selecting new particles from the state transition probabilities $p(x_t|x_{t-1})$ as follows:

$$\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t|\hat{\mathbf{x}}_{t-1}^{(i)}),\ i = 1, \cdots, N. \tag{5.6}$$

In the update stage, the posterior PDF $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is computed by updating the weight of each particle with the likelihoods as follows:

$$w_t^{(i)} \approx p(\mathbf{z}_t|\mathbf{x}_t^{(i)}), \tag{5.7}$$

Subsequently, the posterior probabilities $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ can be approximated through a resampling step as follows:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \Leftrightarrow \{\hat{\mathbf{x}}_t^{(i)}, \frac{1}{N}\}_{i=1}^N. \tag{5.8}$$

5.4 CMIL Tracker with Particle Filter based Motion Prediction

The proposed approach provides a framework that couples a CMIL tracker and a particle filter for robust tracking of articulated human movements while the entire body is executing a large translational motion. Figure 5.2 illustrates the difference between the conventional CMIL tracker and the proposed CMIL tracker with particle filter based motion prediction. The conventional CMIL tracker uniformly samples positive image patches around the estimated position of the target in the previous frame. On the other hand, the proposed method adaptively samples positive image patches at the locations specified by the particle filter. The motion prediction and adaptive sampling made possible by particle filtering provide increased robustness against a large translational motion of the target.

The proposed method is summarized in Algorithm 7. Given, the target estimate from the previous frame, the particle filter first selects new particles using the state transition probability. The particle filter then computes the weight of each particle. For this purpose, we utilize two distance measures, d_B and d_{max}, which we will describe shortly. The posteriori probability is then approximated by a resampling step. Instead of uniformly sampling positive image patches as in the conventional MIL and CMIL, our method collects positive image patches at the locations of new particles using the state transition probability and negative image patches elsewhere. Each of these positive and negative image patches is segmented into a set of components and fed into an online boosting algorithm to update the classifier $\mathbf{H}_t$ [45]. Given the updated classifier $\mathbf{H}_t$, we then compute the feature vector $\mathbf{s}$ of the image patch that corresponds to the estimated position of the target.

We mentioned earlier that we utilize two distance measures for the purpose of computing the weight of each particle. Before we define these two distance measures, we first need to introduce the feature vector $\mathbf{s}$ of an image patch. Recall that each of the positive and negative image patches is segmented into the set of components. The current classifier $\mathbf{H}_t$, when applied to each of these components, produces

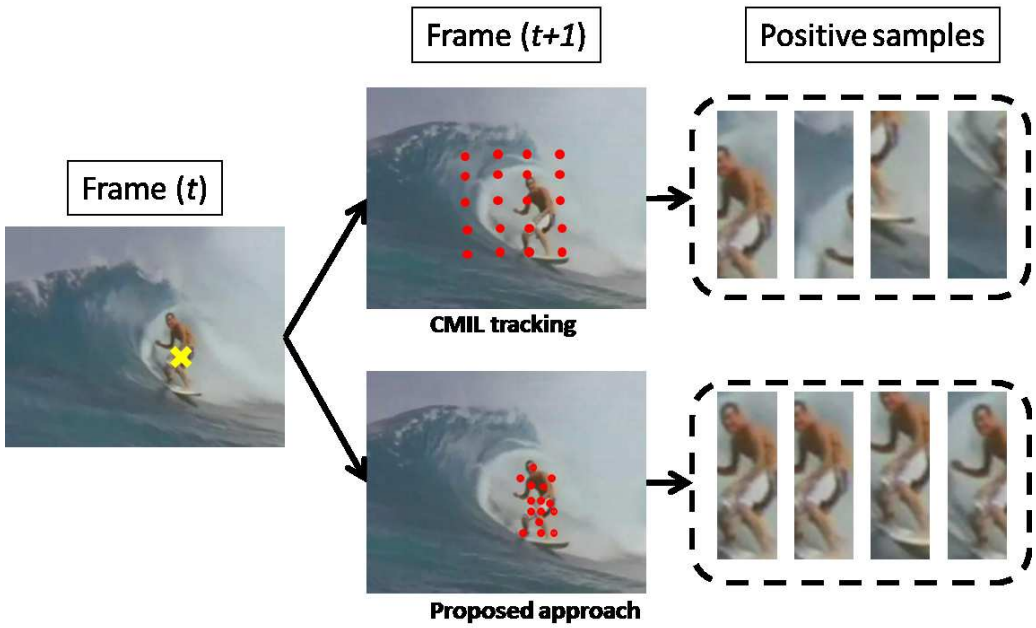


Fig. 5.2. An illustration of the difference between the conventional CMIL tracker and the proposed CMIL tracker with particle filtering. The red dots indicate the sampling positions of a positive image patch at time $t + 1$.

a confidence score of whether that component belongs to a positive instance. Let $\boldsymbol{\nu} = \left\{\nu^{(\ell)} | \ell = 1, \cdots, L\right\}$ be a set of confidence scores where $\nu^{(\ell)}$ is the output of the classifier $\mathbf{H}_t$ applied to the ℓ-th component in the image patch. Based on the set of confidence scores, the feature vector $\mathbf{s}$ is composed of two parts: the first part is the histogram of the confidence score set $\boldsymbol{\nu}$, and the second part is simply the maximum value in the confidence score set. Denoting the histogram as $\mathbf{r}$ and $\nu_{max} \equiv \max(\boldsymbol{\nu})$, the feature vector of an image patch at $\mathbf{x}_t^{(i)}$ is defined as:

$$\mathbf{s}(\mathbf{x}_t^{(i)}) = [\mathbf{r}(\mathbf{x}_t^{(i)}), \nu_{max}(\mathbf{x}_t^{(i)})]. \tag{5.9}$$

Now that we have defined the feature vector of an image patch, we can describe how the weight of each particle is computed. Note that computing the weight of each particle involves computing the similarity between the feature vector of the image patch at the current estimate of the target (i.e., $\mathbf{s}(\hat{\mathbf{x}}_t)$) and the feature vector at each of the particle samples (i.e., $\mathbf{s}(\mathbf{x}_{t+1}^{(i)})$). We use the Bhattacharyya distance [54] as the first distance measure for comparing the two histograms. The Bhattacharyya coefficient measures the degree of overlap between two different discrete distributions, p and q:

$$\rho(p, q) = \sum_{i=1}^{N} \sqrt{p(i)q(i)}, \tag{5.10}$$

and the Bhattacharyya distance is defined as

$$d_B(p, q) = \sqrt{1 - \rho(p, q)}. \tag{5.11}$$

As demonstrated in [44], the maximum confidence score in a positive bag in MIL can be used for estimating the probability of the positive bag . Put it in an equation, we have

$$\Pr(y = 1|\nu) \propto \frac{1}{e^{-\nu_{max}}}, \tag{5.12}$$

where y is the label of bag, $y \in \{-1, 1\}$. The second distance measure between two feature vectors of image patches, therefore, is simply defined as

$$d_{max}(\nu_{max}^{(k)}, \nu_{max}^{(\ell)}) = \nu_{max}^{(k)} - \nu_{max}^{(\ell)}, \tag{5.13}$$

where $\nu_{max}^{(k)} = \nu_{max}(\mathbf{x}^{(k)})$.
Finally, the weight of each particle is then given by

$$\begin{aligned} w_{t+1}^{(i)} &\approx p(\mathbf{z}_{t+1}|\mathbf{x}_{t+1}^{(i)}) \\ &\propto e^{-\gamma(\alpha d_B(\mathbf{r}(\hat{\mathbf{x}}_t), \mathbf{r}(\mathbf{x}_{t+1}^{(i)})) + (1-\alpha) d_{max}(\hat{\nu}_t, \nu_{t+1}^{(i)}))} \end{aligned}, \tag{5.14}$$

where $\hat{\nu}_t = \nu_{max}(\hat{\mathbf{x}}_t)$, $\nu_{t+1}^{(i)} = \nu_{max}(\mathbf{x}_{t+1}^{(i)})$, and γ and α are user-specified control parameters.

5.5 Experiments

We tested our proposed tracker, a CMIL tracker coupled with a particle filter (CMIL-PF), on some challenging video sequences: *Gym*[1], *Skating*[2] , *StandToSit*[3], and *Wiggle*[4] sequences. These sequences include extensive articulated human movements such as bending, rolling, handstand, etc. These sequences also include large translational motions by the human subjects. We compare our algorithm to the online MIL tracker [37] and component-based MIL (CMIL) tracker.

For a fair comparison, all parameters involved in learning the classifiers are fixed in all the trackers and in all the test sequences. In our method, we have fixed the number of particles to be 50 and the noise variance of state transition to be 7 pixels in all test sequences. We have empirically chosen the parameters in Eq. (7.24) for computing the weight of each particle as $\gamma = 2.5$ and $\alpha = 0.3$ in all test sequences. We analyzed the performance of the trackers both qualitatively and quantitatively.

[1]http://www.youtube.com/watch?v=FJIrHgshB20
[2]http://cs.snu.ac.kr/research/~vtd
[3]http://vision.cs.uiuc.edu/projects/activity
[4]http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/

Algorithm 7 CMIL tracker with particle filtering

Input : $\{\hat{\mathbf{x}}_{t-1}^{(i)}, \hat{w}_{t-1}^{(i)}\}_{i=1}^{N}$,$\mathbf{H}_{t-1}$, and $\mathbf{s}_{t-1} = \mathbf{s}(\hat{\mathbf{x}}_{t-1})$

Output : $\{\hat{\mathbf{x}}_{t}^{(i)}, \hat{w}_{t}^{(i)}\}_{i=1}^{N}$, $\mathbf{H}_t$, and $\mathbf{s}_t = \mathbf{s}(\hat{\mathbf{x}}_t)$

Particle Filter Stage

- Predict Motion $\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t \mid \hat{\mathbf{x}}_{t-1}^{(i)})$
- Compute weight $w_t^{(i)} \approx p(\mathbf{z}_t \mid \mathbf{x}_t^{(i)})$

$= e^{-\gamma(\alpha d_B(\mathbf{r}(\hat{\mathbf{x}}_{t-1}),\mathbf{r}(\mathbf{x}_t^{(i)}))+(1-\alpha)d_{max}(\nu_{max}(\hat{\mathbf{x}}_{t-1}),\nu_{max}(\mathbf{x}_t^{(i)})))}$

- Resample particles

$$\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}_{i=1}^{N} \Rightarrow \{\hat{\mathbf{x}}_t^{(i)}, \hat{w}_t^{(i)}\}_{i=1}^{N}$$

CMIL Stage

- Extract image patch at each $\mathbf{x}_t^{(i)}$
- Run CMIL online boost for updating

$$\mathbf{H}_t = \textstyle\sum_k \mathbf{h}_k(\mathbf{x}_t^{(i)})$$

- Compute the updated target feature $\mathbf{s}(\hat{\mathbf{x}}_t)$

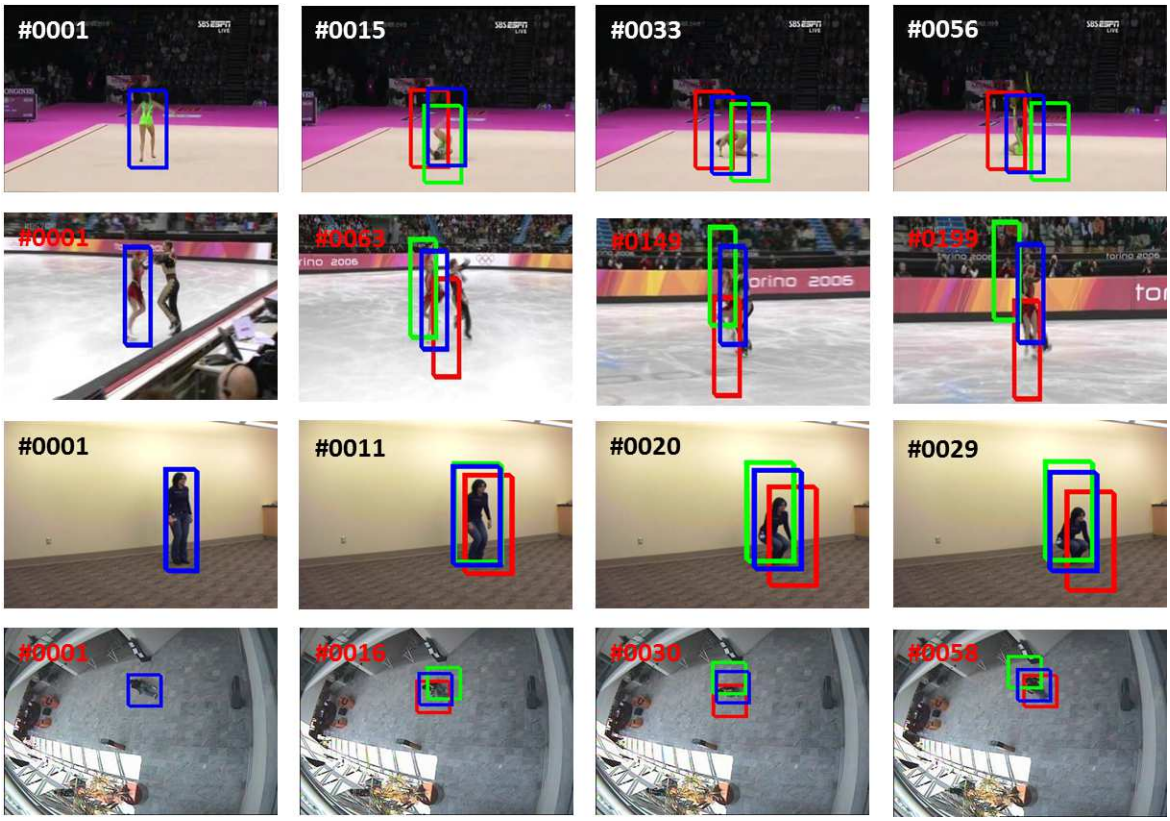


Fig. 5.3. Tracking results on four different test sequences: the test sequences are *Gym*, *Skating*, *StandToSit*, and *Wiggle* from the top to bottom. We compared three different trackers, MIL tracker (Red), CMIL tracker (Green), and CMIL-PF tracker (Blue).

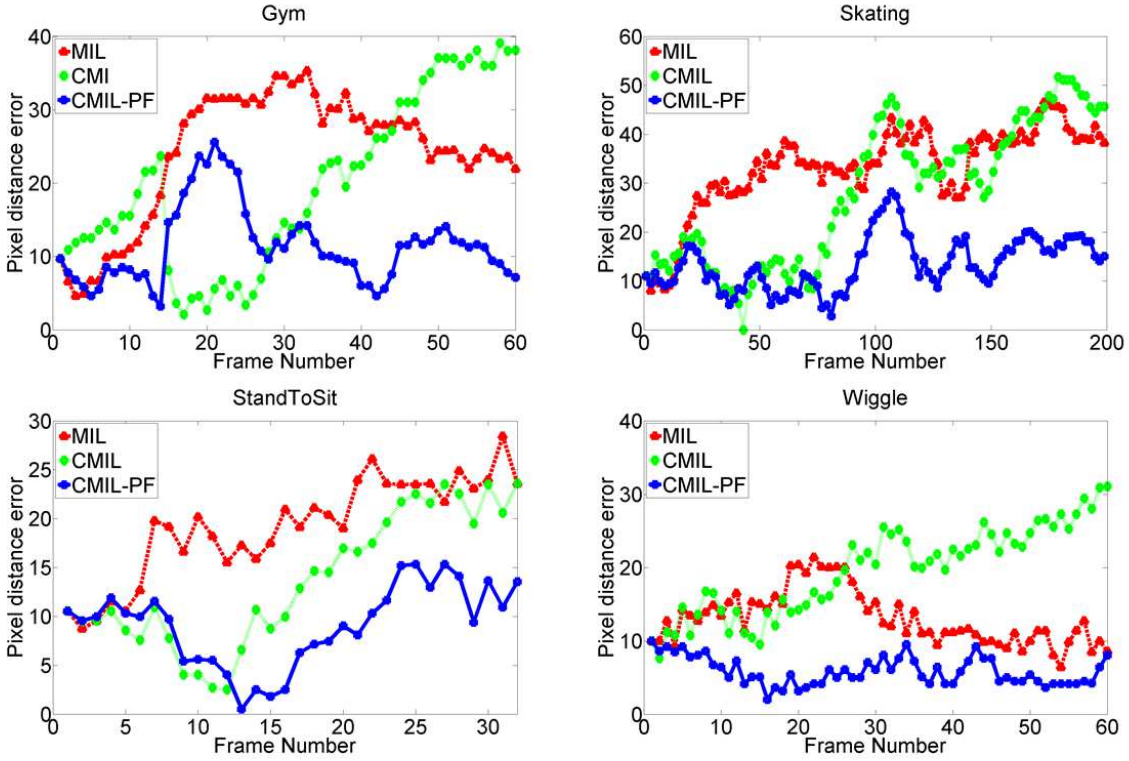


Fig. 5.4. Pixel distance errors on four test sequences.

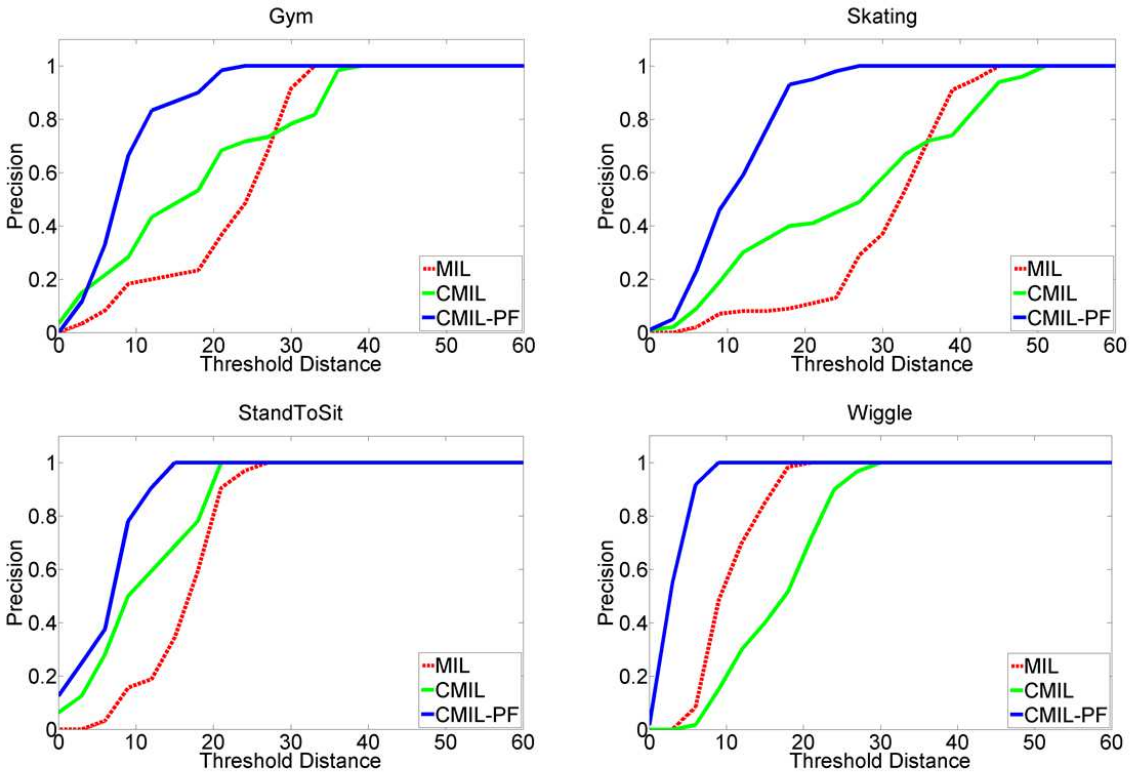


Fig. 5.5. Precision plots of the three trackers on four test sequences.

Figure 5.3 shows the qualitative assessment of the trackers by displaying the bounding box at the position of the human subject estimated by each tracker in each frame. The center position of each rectangle is the best estimated position of the human subject in the image coordinate space.

We also carried out two quantitative evaluations. First, we measured the pixel distance between the target position estimated by each tracker and the ground truth position that are manually collected in all of the test sequences. Figure 5.4 shows the pixel distance errors of the three trackers on each of the four test sequences. In most cases, the proposed CMIL-PF trackers has least pixel distance errors compared to the MIL and CMIL trackers. An exception occurs in the *Gym* sequence at around frame 20 where the CMIL tracker performs better than the CMIL-PF tracker. The exceptional pixel distance errors in these frames are caused by the fact that the human subject is located at the bottom area of the estimated bounding box from the CMIL-PF tracker; in other words, the human subject is not located at around the center of the bounding box. As shown in the second frame (the frame number 15) of the first row in Fig. 5.3 , the center of the bounding box from the CMIL-PF tracker(the blue rectangle) deviates from the center of the human being tracked while the bounding box enclosed the human subject. However, the center of the human subject is located at around the center of the bounding box from the CMIL tracker (the green rectangle). Recall that the target position is the center position of the bounding box, which is used for computing the pixel distance error. Table 5.1 shows the average pixel errors.

The second quantitative evaluation was carried out using precision plots, similar to the ones described in [37]. A precision plot shows the percentage of frames in which the distance between the center position of estimated bounding box and the ground truth position was within some threshold distance. Furthermore, the precision is calculated from frames in which the detected human was located in the bounding box because the error distance is valid only when the detected human is inside the bounding box. Figure 5.5 shows the precision plots of the three trackers on each of

Table 5.1.
Average location error with respect to pixel distance in each test sequence.

	Gym	Skating	StandToSit	Wiggle
MIL track	24.11	33.15	19.19	13.11
CMIL track	19.76	27.94	13.62	19.6
CMIL-PF track	11.44	13.48	9.11	5.84

the four test sequences. The precision plots clearly show that the proposed method increases the tracking accuracy of human subjects with large articulated movements and translational motions. In *Wiggle* sequence, for example, the CMIL-PF tracker was able to track the human subject with the accuracy of 10 pixels or less at all times. Note that only the qualitative result of *Parkour*[5] sequence is shown in Figure 5.1 because the MIL and CMIL trackers lost track within the first couple of frames.

5.6 Conclusion

In this chapter, we addressed the large translational motions of human subject while engaged in articulated movements through the CMIL tracking with the motion prediction framework that is carried out with the particle filter. With motion prediction, the positive image patches in CMIL can be feasibly selected in terms of the probability distribution of human being tracked, so that the classifier in CMIL provides more accurate observation to likelihood computation in the particle filter. We evaluated our proposed tracker by comparing against other existing tracking algorithms in video sequences containing extensive articulated human movements. The proposed tracker showed better tracking performance with respect to both location error and precision in all test sequences.

[5]http://www.youtube.com/watch?v=XuiWzgdA6MA

6. COMPARATIVE STUDY WITH CNN BASED TRACKING

6.1 Introduction

The recent progress in deep learning has tremendously influenced on different technical areas, such as data mining, medical data analysis, natural language processing, semantic scene analysis, and so on. Particularly deep learning frameworks enable us to effectively deal with traditional problems in computer vision, visual object detection and recognition. A convolutional neural network(CNN) – one of the deep learning techniques requiring huge computational power and memory – plays a significant role in the performance of object detection and recognition; specifically improved CNN based image detection and recognition techniques have beaten human benchmark errors [20]. Under the tracking-by-detection framework, an object tracking is one of beneficiaries from this rapid development of deep learning techniques for object detection. The CNN based object tracking [55–64] pursuits to improve tracking performance for different objects, e.g., car, animal, human, etc. The availability of large-scale data in the training stage can realize that CNN based tracking algorithms can successfully track a variety of objects with substantially improved tracking performance. Therefore, it is valuable to conduct a comparative study of the proposed CMIL based tracking algorithms and recent CNN based tracking algorithms for articulated human movement tracking.

From starting to track the target, CMIL based tracking algorithms evolve into more robust tracking models through online training, whereas most CNN based tracking algorithms completely build their tracking models in offline where they are empowered by large-scale training data so that they are able to effectively mature their elaborate tracking models. The proposed CMIL based tracking algorithms could be

probably less comparable to CNN based tracking algorithms because of a shortage of training data and relatively simple model complexity of the CMIL; on the other hand, CMIL based tracking algorithms with online learning, which include no pre-trained information, are likely flexible to unknown target motions than CNN based tracking algorithms even though CNN based tracking algorithms can minimize opportunities of encountering unlearned motions with a wide range of examples in large-scale training data.

Since taking into account an eminence of CNN based tracking algorithms, our primary purpose in this chapter is to investigate achievements of CMIL based tracking algorithms vis-a-vis the state-of-the-art CNN based tracking algorithms rather than whether CMIL based tracking algorithms outperform CNN based tracking algorithms. The comparative analysis presents some details of both CMIL and CNN based tracking algorithms in terms of tracking articulated human movements and we can emplace the proposed CMIL tracking algorithms with respect to the state-of-the-art CNN based tracking algorithms.

The following section 6.2 describes some of recent CNN based tracking algorithms and four selected CNN based tracking algorithms that are used for comparative analysis to the proposed CMIL tracking algorithm. Section 6.3 shows experimental results and analyzes these results to gain insight of CMIL based tracking.

6.2 Related Works

As explained before, CMIL is the learning-based tracking algorithm that requires training data collected during tracking. CNN based tracking algorithm, however, requires large-scale training data to effectively maximize its operability. Therefore, offline training with large-scale data in general is a common way to train CNN based tracking algorithms. Recent CNN based tracking algorithm [62] also supports both offline and online training that allows to update a part of whole CNN tracking mode on-the-fly.

In FCNT [56], the fully convolutional network has been used for tracking the target online manner. The backbone of this tracking network is VGGNet [65]. At the first frame online, FCNT first selects top K features from two feature maps: one contains primitive features, e.g., edge, gradient, etc., and another has semantic features. The subnetwork in FCNT, *GNet*, determines the target location along with removing outliers because the input of GNet is the selected features from the semantic feature map. Otherwise, another subnetwork in FCNT, *SNet*, determines the final target location when there exist a certain number of outliers. During tracking, only SNet is updated to be adaptable target appearance variation and to enhance separability between foreground and background.

SINT [57] builds a VGGNet [65] based Siamese network to train a generic matching function that is robust to common appearance variation in the object. The robustly generic matching function implemented by the Siamese network selects the best matched patch among candidate patches in a new frame with the initial patch from the first frame. The matching function can be realized by sufficient large and diverse training data; therefore, the well-configured training data are inevitably required to obtain improved tracking performances. Irrespective of its reasonable performance, it fails when there are several similar objects in the frame or long occlusion exists.

SiameseFC [59] introduces the Siamese architecture that is composed of fully-convolution layer to generate feature maps. These feature map outputs are integrated to a final heatmap representing the location of object being tracked. Without supporting online model update, SiameseFC refines its outputs by applying a limited search range as well as a cosine window for penalizing large movements. Since SiameseFC can be expected to show possible deficiency to tracking large articulated object movements, e.g., articulated human movements, the comparison to other tracking algorithms reveals its superiority and inferiority with respect to accuracy and robustness, respectively.

Re3 [58] proposes a recurrent network incorporating convolutional layers with supporting multi-level features by skip connections. The convolution layers extract object

appearance information and the recurrent layer supports to store the appearance of object and motion information. Only parameters of the recurrent layers can be updated during tracking. Re3 shows the competitive performance with other compared tracking algorithms in terms of overall performance including accuracy, robustness, and speed.

As a variant of SiameseFC [59], CFNet [60] employs an asymmetric Siamese network that embeds a correlation filter (CF) to which the output feature map of the initial patch of the object will be an input. The CF templates are computed in the training stage. Only difference between SiameseFC and CFNet is to add CF templates optimization into the whole network pipeline. Through combining the Siamese network and CF, the CFNet performs comparatively to the state-of-the-arts tracking algorithm. Especially, CFNet shows superior performance even with shallow depth of CNN because the CF template can compensate the limitation of shallow depth capability.

CREST [55] employs a convolutional residual network to integrate a discriminative correlation filters(DCFs) based tracking scheme. It consists of three layers, a spatial residual layer, a temporal residual layer, and a base convolution layer. The base convolution layer – reformulated DCFs – works for generating spatial correlation response map and the spatial and the temporal layers are operated for compensating the appearance changes of the object being tracked; each layer can be trained online. Authors claim that CREST can alleviate model degradation by incorporating residual networks. CREST reasonably performs on several test data with comparing other state-of-the-arts tracking algorithms. However, in experiments, CREST suffers from blur and rapid motions of the object.

GOTURN [63] consists of an offline learned network. In contrast to most generic object tracking algorithms that supports online training session, the GOTURN proposed the offline learned neural network based generic object tracker at real-time speed.Along with it, instead of classifying many small patches to find the target, it runs a single feed-forward network to regress to the location of target. The two main

components – offline training and single pass regression – enables to track the target at real-time speed.

The network inputs two cropped images from the previous frame and the current frame: one cropped image from the previous frame specifies the target to be tracked and another cropped image for the current frame sets the search range. A sequence of convolution layers outputs high-level features from these cropped images. The outputs of the sequence of convolution layers are then fed into a sequence of fully connected layer that is to find out the target by comparing features of the target from the previous frame and those from the current frame.

Without supporting online training, it can track the target faster than other CNN based tracking algorithms that support online training. Furthermore GOTURN directly estimates only one bounding box so that it does not suffer from increasing computational complexity. However, its performance wholly relies on configurations of offline training data; it may fail to track targets that are far from the offline training data because it has never opportunity to adapt its network by online learning. In addition, during offline training, The network was trained for preferring small motions to large motions under real-world observations. It suffers from irregular motions of the target to be tracked because of its lack of large or irregular motions to be fed into the network during the tracking stage.

DeepSort [64] is a combined framework that performs a Kalman filtering and a CNN based appearance model; the Kalman filtering is to estimate motions of the target and the CNN based appearance model is to extract appearance description of the target. The Kalman filter estimates states of bounding box of the target and the CNN based appearance model extracts appearance descriptors within this estimated bounding box. DeepSort integrates motion and appearance information through a combined metric consisting of Mahalanobis distance and cosine distance, respectively.

DeepSort allows to track multiple persons simultaneously with supporting ID switches during tracking through examining the association metric. It, however, in-

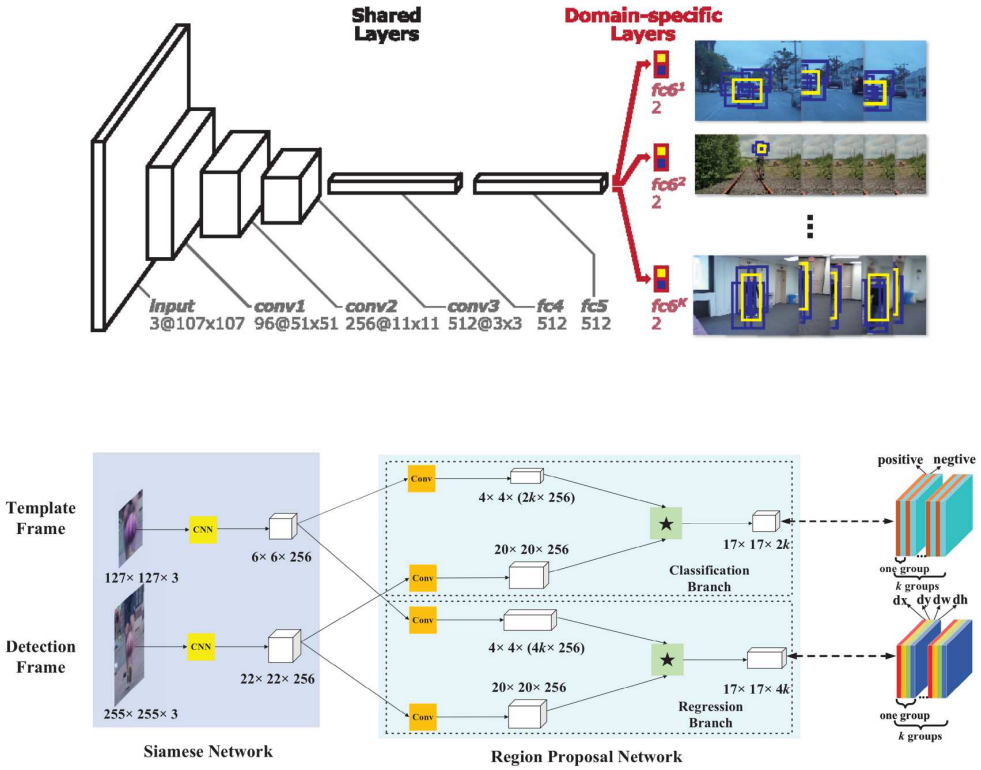


Fig. 6.1. Examples of CNN based tracking framework. In general, input and output of the network are an individual video frame and positions of bounding box surrounding the target being tracked. Reprinted from MDNet [62] and SiamRPN [61].

creases track fragmentation due to sustaining target identities for long-term tracking. Furthermore DeepSort requires external target detectors and currently utilizes fast R-CNN [28]. Therefore, the performance of detector significantly impacts on the overall performance of DeepSORT.

MDNet [62] is designed for building more robust tracking network that is able to overcome a scarcity of variations between targets and background in the training data. MDNet is composed of two types of layers: a shared layer and a number of domain-specific layers. The shared layer pursuits to construct shared representations of targets in the multiple training video data and the domain-specific layer is the a binary classification layer to distinguish the target and background solely. Specifically

the domain-specific layer rephrases the tracking problem as the two-class problem with less computational complexity from its 6 fully-connected layers. Both shared layers and domain-specific layers are trained together with offline training data. During tracking, MDNet creates a new single domain-specific layer and trains it with the shared layer trained offline. Therefore, MDNet is also the online-learning network by creating and updating the new domain-specific layer. This combination of pre- and online trained network provides more flexible trained network that is more suitable for a given target to be tracked whereas most other CNN based tracking algorithms apply pre-trained network in offline. Yet as similar to other CNN based tracking algorithms the performance of network is dependent on a diversity of training data for the shared layer.

SiamRPN [61] incorporates a Siamese network and a region proposal network so that it performs a feature extraction and an estimation of the position of the target being tracked, respectively. Through a template branch and a detection branch in the Siamese network, the SiamRPN encodes the appearance information of the target that are an input of a region proposal network. This region proposal network includes a classification branch and a regression branch; the classification results in the foreground and background classification and the regression branch works for extracting proposal of the bounding box of the object to be tracked. For tracking, as the output of the template branch is considered as the kernel of the detection, the template branch initialized the kernel in the first frame. After the first frame, only the output of detection branch is the input of the regional proposal network. Since only detection branch runs, SiamRPN can reduce computational power during tracking that consequently supports tracking performance at high-rate speed. On the contrary, SiamRPN can fail to track the target with large changes in size and ratio because it also has no update of kernel during tracking. In addition, the refined process for selecting top proposals from outputs of the region proposal network solely takes into account bounding boxes around the vicinity of previous target position due to an assumption of very small motion between successive frames that constrains the

tracking performance. As authors also indicated, the more training data are required to improve the tracking performance.

As described above, most CNN based tracking algorithms require a massive number of training data sets so as to improve their performance; therefore, the scale of training data plays a key role to reduce tracking errors. In addition, since most of CNN based tracking algorithms only allow off-line training, it inevitably causes tracking error when they need to track the targets which are absent in the training data.

6.3 Experiment

We compared the proposed tracking algorithms including CMIL, CMIL-AC, and CMIL-PF with the MIL tracker as well as selected four state-of-the-art CNN based algorithms including GOTURN [63], DeepSort [64], MDNet [62], and SiamRPN [61] among described algorithms in Section 6.2. The selected CNN based tracking algorithms performed well in most popular data sets [66,67]. Especially MDNet presented the best performance of tracking in VOT 2014 [66]. Among these selected CNN based tracking algorithms, only MDNet supports a partial update of network parameters by online training.

We evaluated these eight(8) tracking algorithms on five(5) test sequences that mostly include articulated human movements. The performance of tracking algorithms is measured by two metrics, precision [37] and accuracy [68].

The precision metric is based on the average distance error which is measured by the pixel distance between a center point of the estimated bounding box and that of the ground-truth bounding box across all frames in a given test sequence. This simple average distance error is sometimes insufficient to evaluate the performance of tracking algorithms when distance errors across all frames are widely dispersed.

To compensate this insufficient measurability, the precision also measures a percentage of frames which the distance error is within certain threshold distance error.

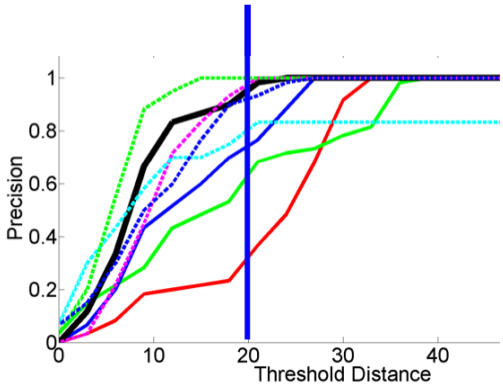


Fig. 6.2. An illustration of the precision with respect to different distance errors. The intersection point of the threshold line(blue) and a precision line(red) represents that 30% of frames have below 20 of distance error.

Figure 6.2 visualizes the 0.3 of precision on 20 threshold distance error that is represented by a point of intersection of a blue straight line with a red curve, which means that 30 percents of frames are less than 20 of distance error.

The accuracy metric is the mean intersection over union (mIOU) that measures the ratio of overlapping area of two bounding boxes.

We used the pre-trained model for each CNN based tracking algorithm. The GOTURN was pre-trained with ALOV300++ data set [69], which is a collection of 314 video sequences, and a set of still images from ImageNet Detection Challenge [70], a total of 134,821 images. The DeepSort [64] was pre-trained with a large-scale person re-identification dataset [71] that contains over 1,100,000 images of 1,261 persons. The MDNet [62] was pre-trained with the ImageNet-VID training set [20] because authors reported that the performance of MDNet with VOT is inferior to that with ImageNet-VID. For SiamRPN [61], the Siamese network is a modified AlexNet trained with ILSVRC [20] and YouTube-BB [72]; the template and the detection branches are trained by patches extracted from two frames of the same video.

Overall, MDNet shows superior tracking performance in OTB and VOT 2014 datasets. Specifically, for VOT 2014 test, MDNet has the top score and ranks with

respect to accuracy and robustness metrics which are corresponding to the bounding box overlap ratio and the number of failures, respectively.

On VOT2015, VOT2016, and VOT2017, the SiamRPN shows the best tracking results with respect to expected averaged overlap (EAO). However, especially for VOT2016, the SiamRPN ranks the 3rd in failure.

6.3.1 Experimental Results and Discussion

As shown in Table 6.1 and Table 6.2, the state-of-the-art CNN based tracking algorithm, SiamRPN [61], shows the best tracking performance. Among CMIL based tracking algorithms, CMIL-PF outperforms other CMIL based tracking algorithms on all test sequences except for *Wiggle* sequence where CMIL-AC has the best performance. Overall, CMIL-PF is comparative to compared CNN based tracking algorithms on all test sequences in terms of the average distance error and precision. In addition, CMIL-PF shows the competitive accuracy even with its shortcomings of both fixed size and ratios of bounding box that likely produce some noises in online training data, while CNN based tracking allows to change the size and ratio of bounding box. In terms of accuracy which a measurement of overlapped regions between the ground-truth bounding box and the estimated bounding box, CNN based tracking more likely conducts better performance. In Table 6.1 and Table 6.2, **Bold** and underline demonstrate the best CMIL based tracking algorithm and the best CNN based tracking algorithm in each test sequence, respectively. The N/A in the Table 6.1 indicates that the corresponding tracking algorithm fails to track the target.

The *Gym* sequence presents that CMIL-PF shows better performance than GOTURN and SiamRPN. The average distance error of CMIL-PF scores 11.44; GOTURN and SiamRPN result in 12.72 and 12.95, respectively. The accuracy of CMIL-PF, 0.41, is closely competitive to that of GOTURN and SiamRPN, both are 0.45. Overall CMIL-PF competes well with CNN based tracking algorithms in *Gym* sequences. Interestingly, both GOTURN and SiamRPN, which are variants of the

Table 6.1.
Comparison between all CMIL and CNN based tracking algorithms in terms of average distance error between ground-truth and expected bounding box in test sequences. The **bold** and underline represent the best CMIL based tracking algorithm and CNN based algorithm, respectively.

	StandToSit	**Wiggle**	**Skating**	**Gym**	**TUD**
MIL	19.19	13.11	33.14	24.11	13.55
CMIL	13.62	19.6	27.94	19.76	9.02
CMIL-AC	13.32	**5.83**	16.45	15.97	13.9
CMIL-PF	**9.11**	5.84	**13.48**	**11.44**	**8.57**
GOTURN	6.99	5.18	14.10	12.72	N/A
DeepSort	23.21	N/A	17.78	N/A	7.96
MDNet	11.07	7.94	10.53	8.40	6.25
SiamRPN	7.65	5.37	8.78	12.95	6.01

Table 6.2.
Comparison between all CMIL and CNN based tracking algorithms in terms of accuracy in test sequences. The **bold** and underline represent the best CMIL based tracking algorithm and CNN based algorithm, respectively.

	StandToSit	**Wiggle**	**Skating**	**Gym**	**TUD**
MIL	0.44	0.57	0.30	0.25	0.63
CMIL	0.53	0.30	0.25	0.25	0.64
CMIL-AC	0.58	**0.61**	0.38	0.32	0.41
CMIL-PF	**0.62**	0.58	**0.43**	**0.41**	**0.65**
GOTURN	0.64	0.61	0.33	0.38	0.17
DeepSort	0.52	0.31	0.41	0.45	0.71
MDNet	0.69	0.69	0.59	0.52	0.77
SiamRPN	0.78	0.71	0.52	0.45	0.76

Siamese network that mainly is designed for learning its parameters with similarities of adjacent frames, result in lower precision scores. Since the *Gym* sequence consists of very severe changes of shape and appearance of the human target by articulated human movements which come from postures of gymnastics, e.g., upside down head to leg position, occlusion of head behind torso, etc. Both GOTURN and SiamRPN suffer from insufficient articulated human movement in their training data from popular training dataset [20, 69, 70, 72]. The CMIL-PF works well for articulated human movements irrespective of limited training data from its online learning framework.

In *StandToSit* sequence, CMIL-PF also is competitive to MDNet with 9.11 and 11.07 of average distance error, respectively. The accuracy of CMIL-PF is relatively not superior to other CNN based tracking algorithms, since the *StandToSit* sequence as well contains continuous changes of size and ratios of human target to be tracked.

It is interesting that all CNN based tracking algorithms except for GOTURN mark high precision and accuracy in the *TUD* sequence. One feasible analysis on this observation is that configurations of training data could have impacted on tracking performance. All CNN based algorithms were pre-trained with large-scale of training data in offline which are configured with different categories of objects and ordinary human motions. The *TUD* sequence consists of a typical human motion, walking pedestrians, for which all CNN based tracking algorithm have been sufficiently learned from large-scale training data.

The test sequence, *Wiggle*, was captured by a top-down and high-angle view of the camera that is mounted on a wall or a ceiling where the shape of the object usually can be distorted by different perspective. Besides the sequence includes a man who lays down first, stands up, and walks away; therefore a whole human body has different shape and appearances – unusual ratios of size between human body parts – across the sequence due to its high-angle view. CMIL-PF favorably performs with 5.84 of average distance error as compared to GOTURN and SiamRPN with 5.18 and 5.37, respectively. Furthermore, CMIL-PF accomplishes better precision than MDNet with 7.94 of average distance error.

The *Skating* sequence includes partial and complete occlusions from interactions between two human bodies. MDNet and SiamRPN accomplish 10.53 and 8.78 of average distance error, respectively and CMIL-PF scores 13.48. This minor performance of CMIL-PF is mainly caused by its online training data that can deteriorate its detection model when the occlusion of the target happens. Once the model is deteriorated, CMIL-PF strives to recover its model with subsequent online training data; hence the recovery of model principally rests on whether the accurate online training data can be collected afterward. CMIL-PF and GOTURN perform closely 13.48 and 14.10 of average distance error, respectively; however, CMIL-PF outperforms GOTURN in terms of accuracy with 0.41 and 0.38, respectively.

Figure 6.6 and 6.7 demonstrates tracking results from compared tracking algorithms on test sequences. As illustrated in Figure 6.6, DeepSort fails to track the target in *Gym* and *Wiggle* sequences. GOTURN is also unsuccessful to track the target in *TUD* sequence as shown in Figure 6.7.

Since we observe tracking performance of each compared algorithm on test sequences which are particularized with articulated human movements, we examine circumstances in which CMIL based tracking algorithms can favorably be on par with the state-of-the-art CNN based tracking algorithms or sometimes outperform them. As reported in Table 6.1, CMIL-PF is likely competitive to CNN based tracking algorithms in terms of average distance error of position of bounding box. With regard to precision, CMIL-PF accomplishes fairly decent tracking performance at cost of limited online training data to both SiamRPN and MDNet that exploit large-scale training data as shown in Figure 6.5. It is important to notice that CMIL-PF is more competent to track the human target with severely articulated movement and perspective distortion than CNN based tracking algorithms; Figure 6.3 depicts *Gym* and *Wiggle* sequences that illustrate these conditions. The *Gym* sequence comprehends strongly noticeable articulated human movements including forward bending torso, forward and side rolling, partial occlusions between body parts, etc. The average distance errors are 11.44, 12.72, and 12.95 in CMIL-PF, GOTURN and SiamRPN, respec-

Fig. 6.3. Examples of circumstances in which CMIL based tracking algorithms are comparable to CNN based tracking algorithms. The first row displays *Gym* sequence including articulated human movement such as bending, rolling, and self-occlusions. The second row illustrates *Wiggle* sequence containing top-down view to cause perspective distortion.

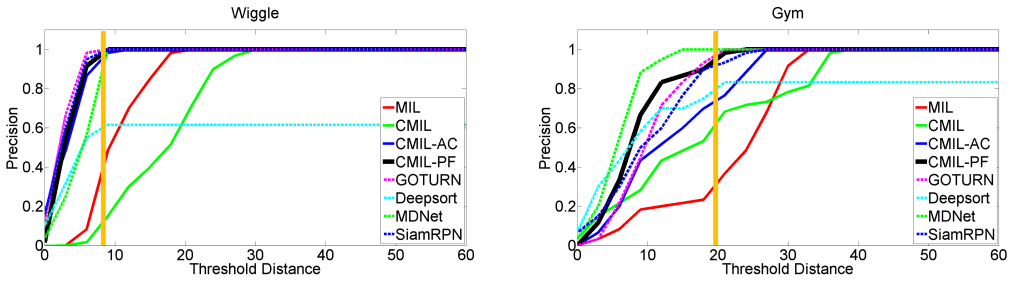


Fig. 6.4. Precision plots for *Wiggle* and *Gym* test sequences on which CMIL-PF is competitive to CNN based tracking algorithm. The yellow vertical line represents the threshold distance which distance errors of all frame are less than.

tively. The accuracy scores are 0.41 and 0.45 in MIL-PF, GOTURN and SiamRPN, both mark the same score, respectively. Likewise, the *Wiggle* sequence is compiled with high-angle view images that lead to peculiarly perspective distortion between human body parts, e.g., the size of head is much larger than that of legs. The average distance errors of CMIL-PF is 5.84 and that of GOTURN and SiamRPN are 5.18 and 5.37. Interestingly the CMIL-PF has better average distance error than MDNet with 7.94. Figure 6.4 illustrates precision plots of *Gym* and *Wiggle* with specific threshold distances. In *Gym* sequence, CMIL-PF, MDNet, and GOTURN reach to 1.0 of precision at 20 of distance error whereas SiamRPN and DeepSort are still below 1.0 of precision. In *Wiggle* sequence, all CNN based tracking algorithms except for DeepSort and CMIL-PF accomplish 1.0 of precision at 9 of distance error. Consequently, CMIL-PF could demonstrate its competitive performance to the state-of-the-art CNN based tracking under such circumstances of severely articulated human motion and perspective distortion as shown in *Gym* and *Wiggle* sequences, notwithstanding its fixed bounding box and restricted online training data.

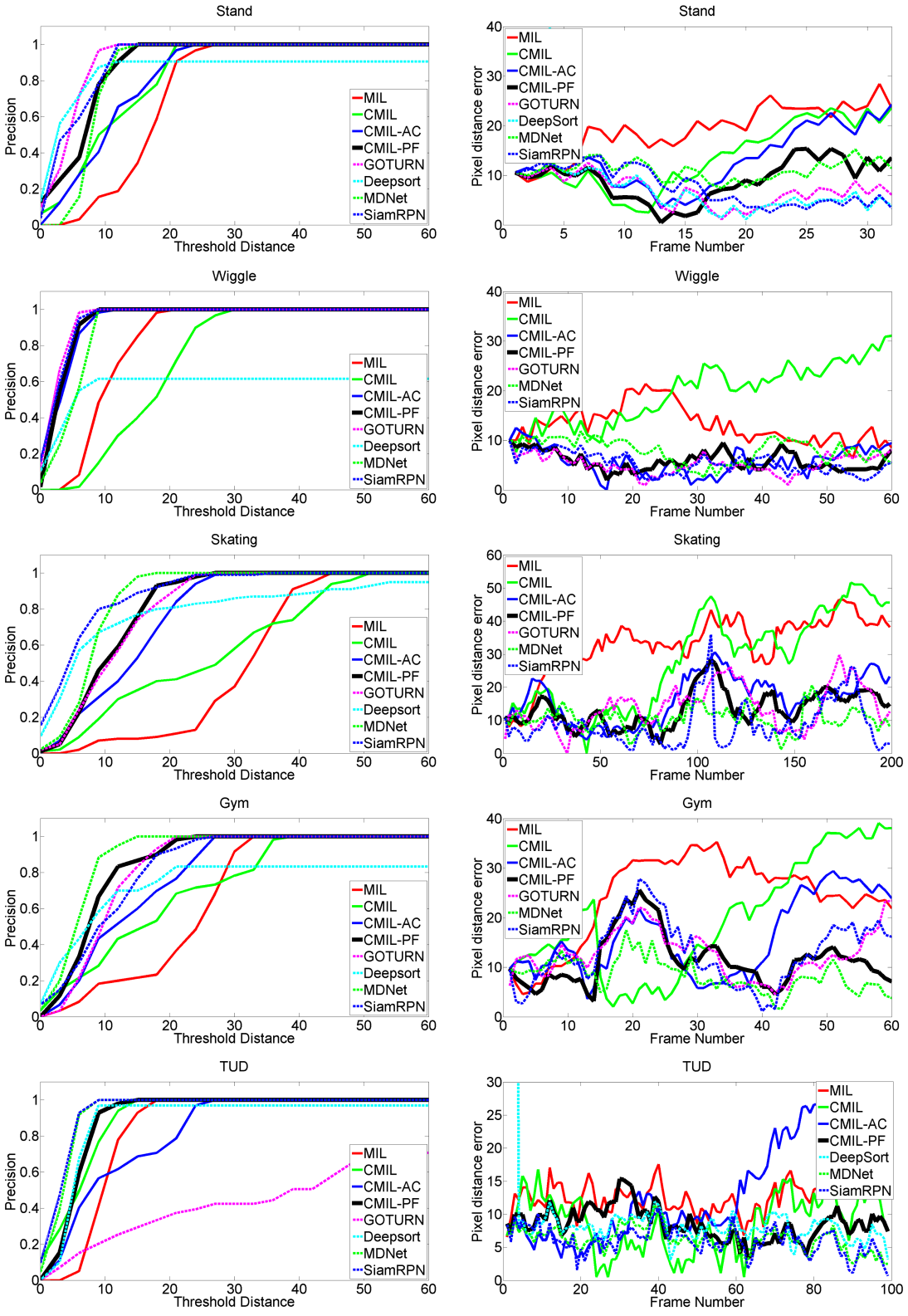


Fig. 6.5. Comparison of precision and distance error across frames. The left column and the right column display the precision plot and the distance error of each test sequence, respectively.

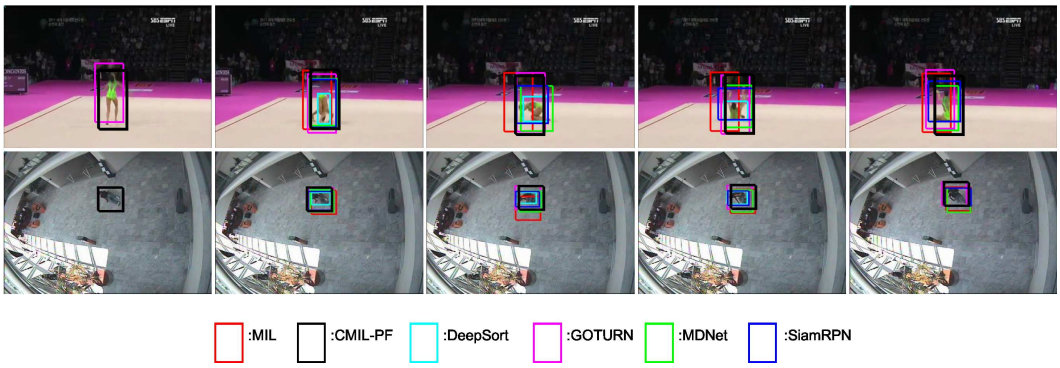


Fig. 6.6. Comparison of estimated bounding boxes from compared tracking algorithms on test sequences. From top to bottom, tracking results are from the *Gym* and *Wiggle*. Note that DeepSort fails to track the target in both sequences

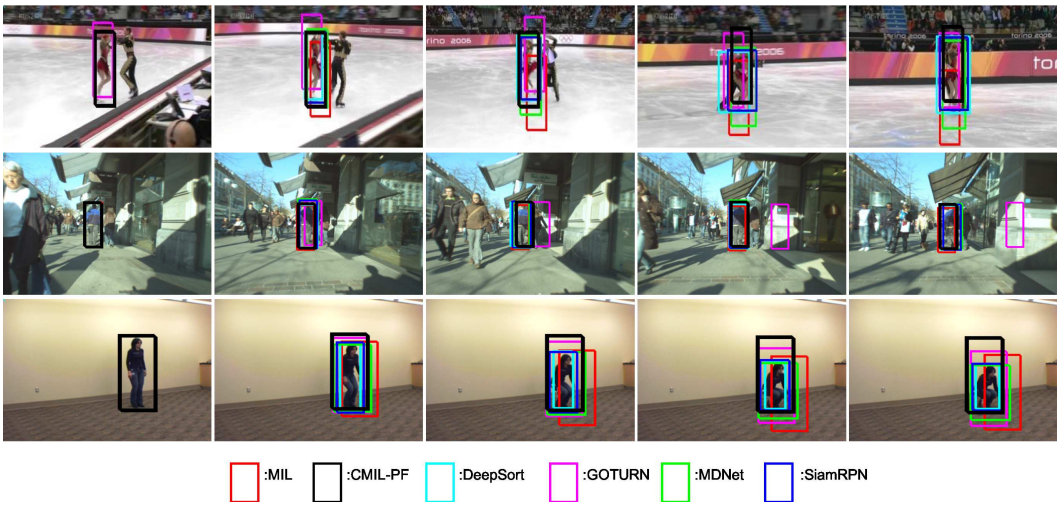


Fig. 6.7. Comparison of estimated bounding boxes from compared tracking algorithms on test sequences. From top to bottom, tracking results are from the *Skating*, *TUD*, and *StandToSit*. GOTURN fails to track the target in *TUD* sequence.

7. GROUND TARGET LOCALIZATION AND TRACKING WITH CMIL FOR UAV VISION

This chapter explains how to apply CMIL for ground target localization and tracking for unmanned aerial vehicle(UAV) vision. The first section introduces a static ground target localization with respect to the ground coordinate space in well-structured environment. The line matching technique is used for refining homography between UAV images and the ground coordinates space, *image-to-ground homographies*. Furthermore, the homography between consecutive frames, what is called *interframe homographies*, is also introduced to estimate the image-to-ground homography of the current frame through propagating the image-to-ground homography in the previous frame. The second section presents the moving ground target localization and tracking in unstructured environment. The CMIL with particle filtering operates on tracking the human subject in the image coordinate space. The interframe homographies are used for both building motion prediction model in the particle filtering framework and estimating the image-to-ground homographies.

7.1 Ground Target Localization Framework in Structured Environment

The goal of ground target localization is to determine the center-of-mass position of one or more ground targets in a world coordinate system using vision-enabled UAVs. Given input aerial images, ray-tracing is widely used owing to its limited time and computational requirements [73–78]. However, ray-tracing has poor accuracy in practice because of the difficulty in estimating the orientation of UAVs. More accurate target localization can be obtained by registering the aerial images to a geo-referenced image provided by a Geographic Information System (GIS) database [78–81].

In general, images of the same outdoor scene which are taken by different cameras and under different conditions can be substantially different. Because of this, feature based image matching is more suitable than direct image matching [82]. We use lines as our basic features for matching since line extraction is invariant to image resolution as well as modest changes in illumination and contrast. Also, lines capture the shape of man-made structures well and they can be stored and manipulated efficiently since they are defined by just their endpoints. Consequently, in this report, we focus on how to register aerial images to the reference image under structured environment where mainly consists of man-made structures rather than natural structure, such as forest, lake, mountains, meadows, etc.

One common approach for line matching is to subsample the lines to obtain points and then apply the Iterative Closest Point (ICP) algorithm which iteratively solves for the transformation between two point sets. ICP has typically been applied to UAVs and Unmanned Ground Vehicles (UGVs) using range sensors, such as Light Detection and Ranging (LIDAR) or sonar. Most of these studies have focused on vehicle state estimation through scan matching. Sappa et al [83] proposed a variant of the ICP algorithm based on edge points in images which used a subsampling technique to reduce the number of outliers and the computational requirements. Hsu et al [84] used ICP in a coarse-to-fine approach which matched two sets of 3D LIDAR data to detect occluded targets. Madhavan et al [85] adapted the weighted ICP for registering two different LIDAR datasets from UGVs. Also, they provided a hybrid registration consisting of feature-based matching followed by point-based matching. Their approach matches LIDAR data from UAVs to data from a UGV. In the context of indoor SLAM using micro UAVs, Sober and Johnson [86] used a laser sensor and sonar to augment an inertial measurement unit. The ICP algorithm is generally robust against noise and missing information and is guaranteed to have decreasing error in every iteration and hence a local minimum (but not necessarily the global minimum) is always reached [87]. Unfortunately, processing on UAVs is time-bounded, therefore convergence may not be possible within the allotted time.

To address the time-bounded problem, we bail out of ICP after a fixed number of iterations. This breaks the convergence guarantees of ICP. However, we show that by propagating information across subsequent frames we can still achieve desired convergence. The key observation is that the geo-registration solutions for two subsequent frames are related by the homography between the frames. By computing this homography, we can propagate the partial ICP results into the next frame and thus still achieve convergence as long as the error reduction in each frame is greater than the error introduced by the interframe homography. Our experimental results support our claim, with a substantial improvement in ground target localization after a few frames compared to backprojection.

A second issue with ICP is that the algorithm only guarantees convergence to a local minimum [88] and hence there is no guarantee that it will give the correct geo-registration. This problem with ICP can be solved by restarting the algorithm from a number of different initial positions and keeping the minimum cost solution [89]. Another solution is to run ICP several times using randomly selected subsets of the points and again keeping the minimum cost solution [90]. Neither solution is feasible for real-time processing on a UAV.

Fortunately, we can increase the likelihood that ICP converges to the correct geo-registration by simply using the ICP results from the previous frame to initialize ICP in the next frame – by the aforementioned homography between two subsequent frames. Each new frame captured by the UAV reveals some new data which was not present in the previous frame. If the geo-registration in the previous frame was correct, then this new data will agree closely with the initialization and ICP will again converge to the correct solution. On the other hand, if the geo-registration in the previous frame was not correct, then the new data is unlikely to agree with the initialization and ICP will converge to a new solution. The instability of incorrect solutions and the stability of the correct solution makes it likely, though not guaranteed, that the correct solution will be achieved given enough frames. This intuition is borne out in our experimental results which show that ICP sometimes requires a few frames to

converge to the correct registration, but once the correct registration is achieved it is maintained for as long as the interframe homographies can be correctly estimated.

The high level structure of our proposed method for geo-registration aerial images shown in Fig. 7.1. As noted above, our strategy for localizing ground targets is to determine the geo-registration of each input image which has been previously shown to be an effective method [79–81]. The novelty in this report lies in our use of line features along with ICP in order to obtain a quick and robust registration. Line features are largely invariant to illumination and contrast changes. Furthermore, by using lines we do not require the images to be of the same type, e.g., the reference GIS database could be made up of geo-referenced optical or IR images or even floor plans for man-made structures in the area. Our geo-registration algorithm is made up of two major steps: (1) extract parameterized lines from the aerial image and (2) match these lines to a set of reference lines using ICP.

7.1.1 Line Extraction in Aerial Images

We extract lines by first applying the Canny edge detector [91] and then organizing the low-level edge pixels into higher-level line features using the k-adjacent segment algorithm [92]. We then remove spurious lines – lines whose length is shorter than a pre-specified threshold or lines that are connected by an acute angle – in order to produce a set of lines which are more likely to correspond to the artificial landmarks in the image (e.g., man-made structures such as buildings and roads). By pruning the lines in this way we reduce the computational requirements of the system and also increase the robustness of the matching process.

An example of the extracted lines before and after pruning the spurious lines is shown in Fig. 7.2.

Before proceeding, we need to transform the data into the camera coordinate system. We map each endpoint of each line (x_i, y_i) to the corresponding camera coordinates (x_c, y_c) by following equations.

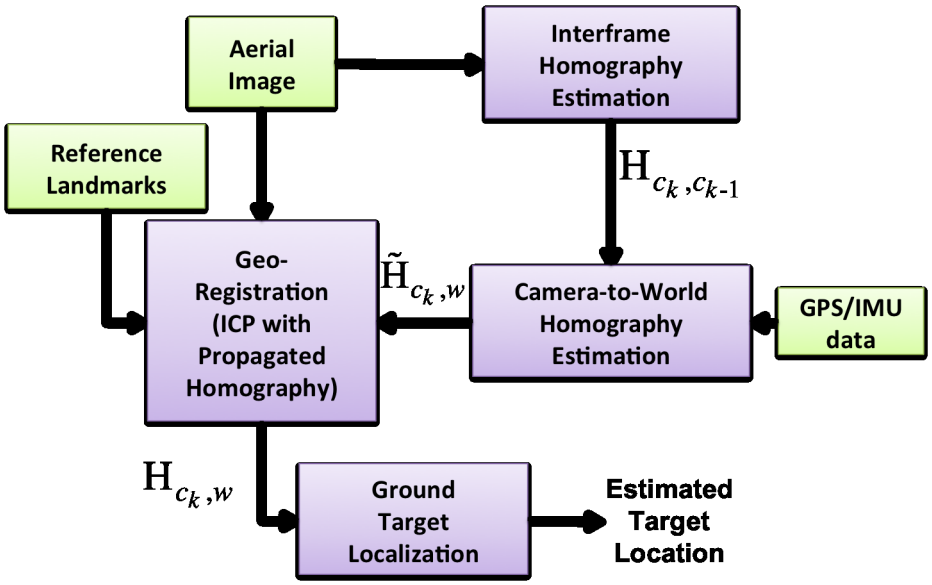


Fig. 7.1. The system structure of ground target localization in structured environment through combination of GPS/IMU information, interframe homography, current aerial image, and the reference image.

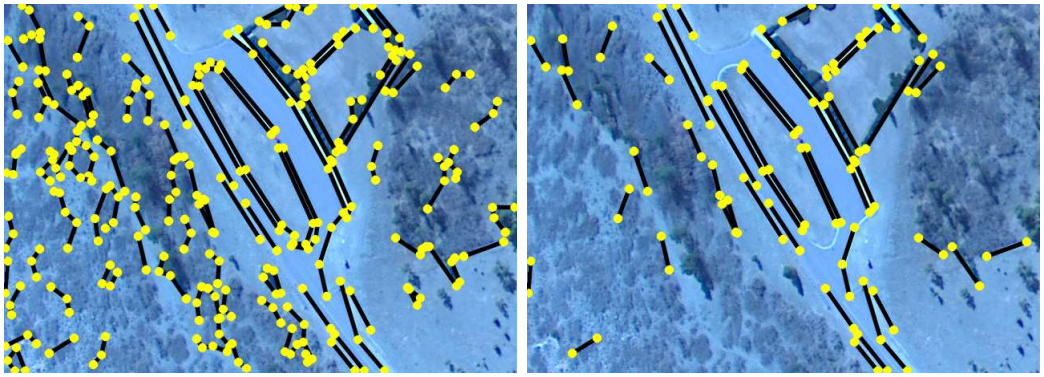


Fig. 7.2. An example of pruning spurious lines: initial extracted lines (left) and refined lines after applying length and angle constraints (right). The example shows the successful removal of spurious lines appeared in unstructured regions.

$$\begin{bmatrix} x_{c,d} \\ y_{c,d} \end{bmatrix} = \begin{bmatrix} (x_i - c_x)/f_x \\ (y_i - c_y)/f_y \end{bmatrix}, \tag{7.1}$$

where f_x and f_y are the focal lengths of the camera and (c_x, c_y) is the principal point. We then correct for radial distortion using

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} \frac{x_{c,d}}{\theta_d} \cdot \tan\left(\frac{\theta_d}{1+\kappa_1\theta_d^2+\kappa_2\theta_d^4+\kappa_3\theta_d^6+\kappa_4\theta_d^8}\right) \\ \frac{y_{c,d}}{\theta_d} \cdot \tan\left(\frac{\theta_d}{1+\kappa_1\theta_d^2+\kappa_2\theta_d^4+\kappa_3\theta_d^6+\kappa_4\theta_d^8}\right) \end{bmatrix}, \tag{7.2}$$

where κ_i is the i^{th} lens distortion coefficient and $\theta_d = \sqrt{x_{c,d}^2 + y_{c,d}^2}$.

7.1.2 Image Registration via ICP

The iterative closest point (ICP) algorithm is a method for aligning two point clouds, first introduced by Besl and McKay [88]. Given points $\mathbf{p}, \mathbf{q} \in \mathbb{R}^{\mathbf{2}}$, a set of data points $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_{n_p}\}$ and model points $M = \{\mathbf{m}_1, \ldots, \mathbf{m}_{n_M}\}$, the goal of ICP is to compute a rotation matrix R and a translation vector $\mathbf{t} \in \mathbb{R}^2$ such that the transformed data points are best aligned with M. The following is a summary of the algorithm:

Fig. 7.3. Two other examples of aerial images (top row) and the extracted lines in those image (bottom row); the refined lines, i.e., excluding spurious lines, are appeared at boundaries of buildings or roads. It indicates that line features are likely suitable for representing structured environment.

1. Initialization:$k = 0$, $\mathrm{R} = \mathrm{I}_{2\mathrm{x}2}$, $\mathbf{t} = \mathbf{0}$.

2. Compute the closest point: For each point in $\hat{P} = \mathrm{R}P + \mathbf{t}$, compute the closest point in M subject to a set of conditions (note that some points in $\hat{P}$ may not be matched if no points in M meet the conditions). The result is a set of m correspondences $\mathbf{p}_i \leftrightarrow \mathbf{m}_{c(i)}$.

3. Compute the registration: Given the set of corresponding pairs, compute a new R and $\mathbf{t}$ which minimize $J(\mathrm{R}, \mathbf{t}) = \frac{1}{Z}\sum_{i=1}^{Z} \left\|\mathbf{m}_{c(i)} - \mathrm{R}\mathbf{p}_i - \mathbf{t}\right\|_2$.

4. Check for convergence: Terminate if the reduction in cost is below a threshold or if a maximum number of iterations has been reached.

The final homography is given by $\mathrm{H}_{icp} = \begin{bmatrix} \mathrm{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$.

In order to match line features extracted from aerial images to the GIS database, we must first apply two preprocessing steps to the data. The ICP algorithm is limited to estimating a similarity transform and so we transform the endpoints of the extracted lines based on an initial estimate of the image-to-ground homography $\tilde{\mathrm{H}}_{c_k,w}$ with the assumption that this estimate mostly accounts for any perspective distortion. Our method for determining $\tilde{\mathrm{H}}_{c_k,w}$ is described in the next section. Since the ICP algorithm is based on point matching, we sample each line at 1 meter intervals. This is done for both the transformed lines from the aerial image as well as the lines from the GIS database. The sample points from the aerial image become the set P in the above algorithm while the sample points from the GIS database become the set M.

For the matching in step 2, we define the closest point to $\mathbf{p}_i \in P$ to be

$$\begin{aligned} \mathbf{m}_{c(i)} \quad &= \quad \underset{\mathbf{m}_i \in M}{argmin} \|\mathrm{R}\mathbf{p}_i + \mathbf{t} - \mathbf{m}_j\| \\ subject\,to \quad & \quad \|\mathbf{p}_i - \mathbf{m}_j\| \leq d_{max} \\ & \quad \frac{\left|\mathrm{R}\mathbf{n}_{\mathbf{p}_i} \cdot \mathbf{n}_{\mathbf{m}_i}\right|}{\|\mathbf{n}_{\mathbf{p}_i}\| \|\mathbf{n}_{\mathbf{m}_i}\|} \geq \cos(\theta_{max}) \end{aligned} \quad ,$$

where $\mathbf{n_x}$ is the normal vector to the line passing through the point $\mathbf{x}$. The maximum angle threshold θ_{max} ensures that only points from lines which roughly agree on orientation are matched. The maximum distance threshold d_{max} is intended to remove outliers during the matching process. Because of this, it is important to start with a large value for d_{max} and then progressively reduce it with each iteration. The initial maximum distance threshold d_{max} is set and updated by

$$d_{max} = \begin{cases} \max\left\{D_{th}, \mu_d + \lambda\sigma_d\right\}, & \text{first iteration} \\ \\ (1-\alpha)\eta_d + \alpha\xi_d, & \text{otherwise} \end{cases}, \tag{7.3}$$

where D_{th}, λ, and α are user parameters, μ_d and σ_d are the mean and standard deviation of the matching distances from the final iteration of ICP in the previous frame, and η_d and ξ_d are the mean and max distances respectively from the preceding iteration.

By expressing d_{max} in this way, the distance threshold is always between the mean and max distance from the preceding iteration for any $\alpha \in [0, 1]$. Once the ICP algorithm converges or reaches the maximum number of iterations we compute the final estimate of the image-to-ground homography

$$\mathrm{H}_{c_k,w} = \mathrm{H}_{icp}\tilde{\mathrm{H}}_{c_k,w}, \tag{7.4}$$

where $\tilde{\mathrm{H}}_{c_k,w}$ is an initial estimate of the image-to-ground homography. Once we have computed $\mathrm{H}_{c_k,w}$ it is straightforward to convert any $\mathbf{x}_i$ in the image coordinate system (e.g. the image coordinates of a target) to the corresponding $\mathbf{x}_w$ in the world coordinate system as follows:

$$\mathbf{x}_w = \mathrm{H}_{c_k,w}\,\mathbf{x}_c, \tag{7.5}$$

where $\mathbf{x}_c$ is the corresponding point of $\mathbf{x}_i$ in the camera coordinate system.

7.1.3 Initializing ICP Using Interframe Homographies

In the previous section, we discussed how ICP can be used to improve the initial estimate of the image-to-ground homography $\tilde{\mathrm{H}}_{c_k,w}$. In this section we discuss how to obtain this estimate.

The simplest way to obtain the estimate $\tilde{\mathrm{H}}_{c_k,w}$ is to use GPS/IMU information. Given the estimated position and orientation of the UAV,

$$\mathrm{Y} = \begin{bmatrix} cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{7.6}$$

$$\mathrm{P} = \begin{bmatrix} cos(\beta) & 0 & sin(\beta) \\ 0 & 1 & 0 \\ -sin(\beta) & 0 & cos(\beta) \end{bmatrix}, \tag{7.7}$$

$$\mathrm{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\gamma) & -sin(\gamma) \\ 0 & sin(\gamma) & cos(\gamma) \end{bmatrix}, \tag{7.8}$$

$$\mathrm{H}_{pos} = \begin{bmatrix} 0 & 1 & pos_x/pos_z \\ 1 & 0 & pos_y/pos_z \\ 0 & 0 & 1/pos_z \end{bmatrix}, \tag{7.9}$$

where α, β, and γ represent the yaw, pitch, and roll angles respectively, and pos_x, pos_y, and pos_z are the UAV's x, y, and z positions in the world coordinate system.

The desired estimate for the image-to-ground homography is computed by

$$\tilde{\mathrm{H}}_{c_k,w} = \mathrm{H}_{pos}\,\mathrm{H}_{Rot}\,\mathrm{H}_{c,g}. \tag{7.10}$$

where H_{pos} is determined from the position of the UAV, $\mathrm{H}_{Rot} = \mathrm{Y\,P\,R}$ is determined from the orientation of the UAV, and $\mathrm{H}_{c,g}$ accounts for the extrinsic calibration between the camera and GPS/IMU unit. While this method is fast, it often fails to

give a good estimate due to the difficulty in accurately measuring the position and orientation of the UAV. For example, in our experiments the UAV is at an altitude of 220m. At this altitude, an error in the pitch or roll measurement of just 1° results in a localization error of around 4m if the UAV is directly overhead and much more than 4m if the UAV is viewing the target at an oblique angle. Difficulty in estimating the position of the UAV (particularly the height) also introduces a substantial amount of error.

A comparison of the red and green lines in Figure 7.10 reveals that using ICP with $\tilde{\mathrm{H}}_{c_k,w}$ determined from the GPS/IMU reduces the localization error by a factor of two compared to using ray-tracing. While this is a substantial improvement, much better results can be obtained by using a better initialization.

In our method, we observe that the geo-registration solutions for two subsequent frames are related by the homography between the frames, i.e., the interframe homography. Note that this homography can typically be estimated very efficiently and accurately because the rotation and translation between subsequent frames is limited by the kinematic and dynamic constraints of UAVs.

As illustrated in Fig. 7.4, the initial estimate of the image-to-ground homography of the current frame is given by

$$\tilde{\mathrm{H}}_{c_k,w} = \mathrm{H}_{c_{k-1},w}\,\mathrm{H}_{c_k,c_{k-1}}, \tag{7.11}$$

where $\mathrm{H}_{c_{k-1},w}$ and $\mathrm{H}_{c_k,c_{k-1}}$ are the image-to-ground homography of the previous frame and the interframe homography from the current frame to the previous frame, respectively. Of course, this initialization can only be used when $\mathrm{H}_{c_{k-1},w}$ and $\mathrm{H}_{c_k,c_{k-1}}$ are available, i.e., only after the first frame and only when $\mathrm{H}_{c_k,c_{k-1}}$ can be accurately estimated. Otherwise we fall back on using the GPS/IMU to obtain $\tilde{\mathrm{H}}_{c_k,w}$.

To estimate $\mathrm{H}_{c_k,c_{k-1}}$, we first extract and match Speeded-Up Robust Feature (SURF) features [93] from each frame and then transform the coordinate of each feature point to the camera coordinate system. Finally, we apply Random Sample Consensus (RANSAC) to robustly estimate the homography in the presence of

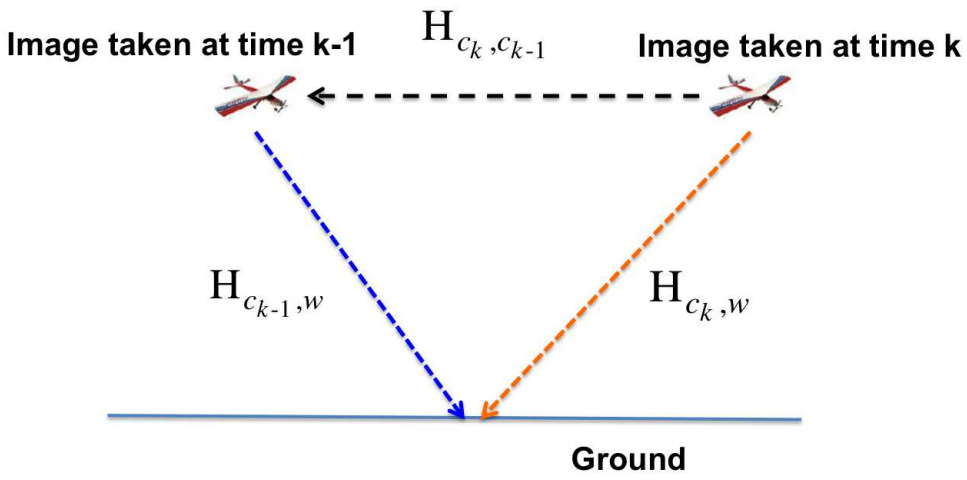


Fig. 7.4. Illustration of the image-to-ground homography of the current frame $H_{c_k,w}$, the image-to-ground homography of the previous frame $H_{c_{k-1},w}$, and the interframe homography $H_{c_k,c_{k-1}}$

outliers. We evaluate the estimated homography via the number of of inliers from RANSAC and the magnitude of the rotation and translation. If the number of inliers is below a threshold or the rotation and translation exceed a threshold then we discard $\mathrm{H}_{c_k,c_{k-1}}$ and use the GPS/IMU to obtain $\tilde{\mathrm{H}}_{c_k,w}$.

Using Eq. (7.11)to initialize ICP can be understood as using the interframe homography to transfer the final ICP results from the previous frame to the current frame. This allows us to overcome two potential difficulties with ICP, the open-ended time requirements due to the fact that it is an iterative algorithm and the difficulty in avoiding local minima.

First, since real-time processing on a UAV is time-bounded, we limit the number of iterations of ICP to a small fixed number. Because of this, ICP is typically not able to converge if the initial error is high. However, by carrying these partial results into the next frame, in a sense ICP is able to continue the iterations. In this way, we still obtain convergence after several frames. Section 7.1.6 gives some experimental results that support the validity of this claim.

Using interframe homographies to initialize ICP also helps to reduce the problem of local minima. Given enough iterations, ICP is guaranteed to converge to a local minimum near the initialization point. In general, there is no guarantee that this local minimum will correspond to the correct geo-registration. However, if $\mathrm{H}_{c_{k-1},w}$ is a correct geo-registration, then using Eq. (7.11) to initialize ICP increases the likelihood of converging to the correct geo-registration. On the other hand, if $\mathrm{H}_{c_{k-1},w}$ is not a correct geo-registration then the initialization obtained using Eq. (7.11) is unlikely to be near the correct local minimum and thus ICP will likely converge to an erroneous geo-registration. In practice, however, given enough frames and a rich enough scene we eventually obtain convergence to the correct geo-registration as shown in Fig. 7.8.

Algorithm 8 Propagated Iterative Closest Point Algorithm

Given:

The reference point set $M = \{m_i\}_{i=1}^{N_M}$.

Do

(1) Given an aerial image, extract line from the aerial image and do sampling points l_i from each line.

(2) Extract SURF points the current aerial image and Compute interframe homography $\mathrm{H}_{\mathrm{c_k},\mathrm{c_{k-1}}}$.

(3) Compute the initial camera-to-ground homography $\tilde{\mathrm{H}}_{c,w}$.

i) if $\mathrm{H}_{\mathrm{c_k},\mathrm{c_{k-1}}}$ passed the evaluation test, the initial homography $\tilde{\mathrm{H}}_{c,w} = \mathrm{H}_{\mathrm{c_{k-1}},\mathrm{w}}\,\mathrm{H}_{\mathrm{c_k},\mathrm{c_{k-1}}}$.

ii) if $\mathrm{H}_{\mathrm{c_k},\mathrm{c_{k-1}}}$ failed to pass the evaluation test, the initial homography $\tilde{\mathrm{H}}_{c,w} = \mathrm{H}_{\mathrm{pos}}\,\mathrm{H}_{\mathrm{Rot}}\,\mathrm{H}_{\mathrm{c,g}}$.

(3) Compute back-projected point p_i from the sample point l_i using $\tilde{\mathrm{H}}_{c,w}$ and construct a set $P = \{p_i\}_{i=1}^{N_P}$.

(4) Run ICP and compute $\mathrm{H}_{\mathrm{icp}}$ from P and M.

(5) Refine the initial camera-to-ground homography $\tilde{\mathrm{H}}_{c,w}$ by $\mathrm{H}_{\mathrm{c,w}} = \mathrm{H}_{\mathrm{icp}}\,\tilde{\mathrm{H}}_{\mathrm{c,w}}$.

while still available aerial images

7.1.4 Uncertainty of Estimated Transformation

We can utilize the covariance of estimated homography parameters in order to compute the covariance of the position of ground targets. For any homography, given two points $\mathrm{m_c}$and p satisfying the transformation m = Hp , the covariance of point m is given by

$$\Sigma_{\mathrm{m}} = J_{\mathrm{h}} \Sigma_{\mathrm{h}} J_{\mathrm{h}}^T, \tag{7.12}$$

where $\Sigma_{\mathrm{h}}$is the covariance matrix of the parameters of the homography $\mathrm{H_{icp}}$ and J_{h} is the Jacobian matrix representing $\frac{\partial \mathrm{m}}{\partial \mathrm{H}}$ [94]

$$J_{\mathrm{h}} = \begin{bmatrix} \mathrm{x}^T & 0^T & -\mathrm{x}'\mathrm{x}^T \\ 0^T & \mathrm{x}^T & -\mathrm{y}'\mathrm{x}^T \end{bmatrix}. \tag{7.13}$$

where $\mathrm{p}_i^T = [\ x_i \quad y_i \quad 1\ ]^T$, $\mathrm{m}_i^T = [\ x_i' \quad y_i' \quad 1\ ]^T$.

Σ_h can be estimated based on the corresponding pairs used to estimate the homography. Given n corresponding pairs $\mathrm{m}_i \leftrightarrow \mathrm{p}_i$ we can estimate the parameters of H by minimizing

$$E(\mathrm{h}) = \sum \|\mathrm{Ah}\|_2 , \tag{7.14}$$

where $A = \begin{bmatrix} \mathbf{0}^T & -\mathbf{p}_1^T & x_1'\mathbf{x}_1^T \\ \mathbf{p}_1^T & \mathbf{0}^T & -y_1'\mathbf{x}_1^T \\ \vdots & \vdots & \vdots \\ \mathbf{0}^T & -\mathbf{p}_n^T & x_n'\mathbf{x}_n^T \\ \mathbf{p}_n^T & -\mathbf{0}^T & -y_n'\mathbf{x}_n^T \end{bmatrix}$ and $\mathrm{h} = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 & h_8 & h_9 \end{bmatrix}^T$.

If we compute the SVD decomposition of A, $\mathrm{A} = \mathrm{UWV}^T$, we can compute Σ_{h} as

$$\Sigma_{\mathrm{h}} = \mathrm{Cov}(\mathrm{h}_{jk}) = \sum_{i=1}^{\mathrm{M}} \left(\frac{\mathrm{V}_{ji}\mathrm{V}_{ki}}{w_i^2} \right), \tag{7.15}$$

where V_i is the i^{th} column of V [95].

7.1.5 Experiments

For evaluating geo-registration of aerial images, we conducted outdoor flight experiments to localize a static ground object on a road using a single UAV shown in Figure 7.5. To generate the reference landmark lines, we first captured a satellite image of the applicable region from Microsoft Bing map and then manually extracted lines on the landmarks. The world geodetic system (WGS) coordinates of two end points of each line were obtained using the Microsoft Bing map API. These WGS coordinates were then mapped into a locally defined world coordinate system. The origin of the world coordinate system was set to the location of a known ground station with $+x$ to the east and $+y$ to the north.

An example aerial image and the static ground target, a red car, are shown in Figure 7.7. The ground truth WGS position of the ground target was manually measured by a GPS device prior to conducting the experiment.

Our dataset consists of 250 aerial images of the region containing the target. For each image, color based segmentation was used to detect the ground target with the center of mass used as the ground target location in the image coordinate system. The extracted lines from each aerial image were then registered to the reference lines using ICP. We set $\alpha = 0.95$, $D_{th} = 10$m, and $\lambda = 3$ in Eq. 7.3 for determining d_{max} in each iteration of ICP. The value of θ_{max} was fixed at 45 degrees for all iterations.

Figure 7.8 shows corresponding pairs between query and model lines for selected iterations of ICP. In Figure 7.8, the yellow square in the image indicates the ground truth target location, the white square shows the estimated target location by ray-tracing using the GPS/IMU information, and the cyan square shows the current estimated target location based on ICP.

Figure 7.10 shows the ground target localization error over the whole dataset for three different methods while TABLE 7.1 summarizes this information. The three methods are: (1) conventional ray-tracing via GPS/IMU information; (2) ICP with the initial image-to-ground homography computed from only the GPS/IMU informa-

(a)

(b)

Fig. 7.5. Unmanned Aerial Vehicles in experiments: (a) UAV and (b) an example image captured in the experiment.

Table 7.1.
Ground target localization error with unlimited ICP iterations

	Ray-Tracing	ICP with GPS/IMU	ICP with interframe homography
median error (m)	60.2964	24.4213	4.2373

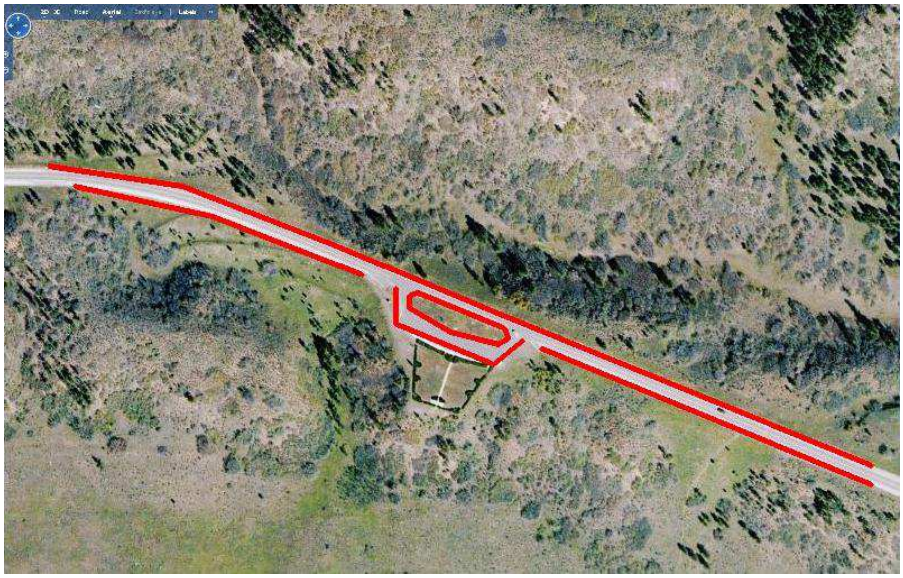


Fig. 7.6. The reference satellite image and the set of reference lines represented by red line; the geodesic coordinate and pixel position of two end points of each reference line have been manually extracted.

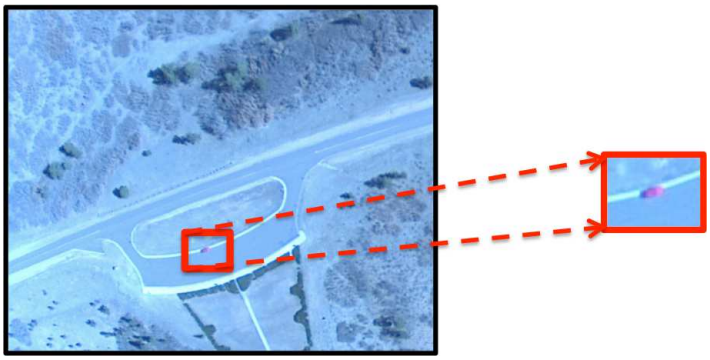
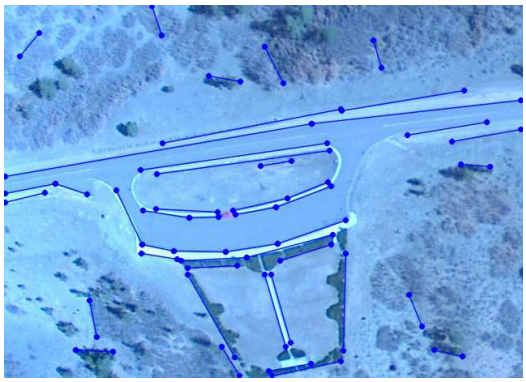
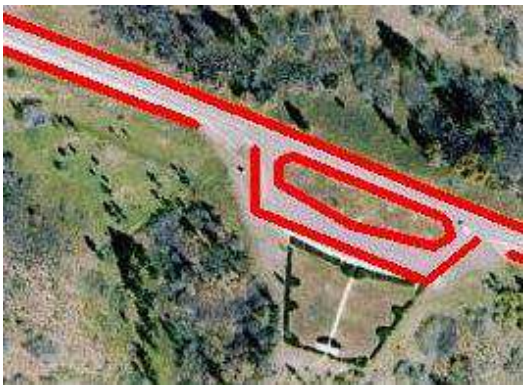


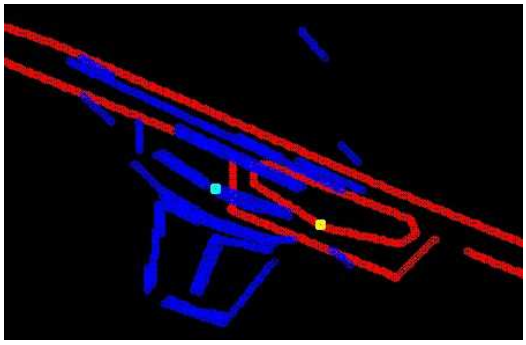
Fig. 7.7. The ground target marked in a red rectangle



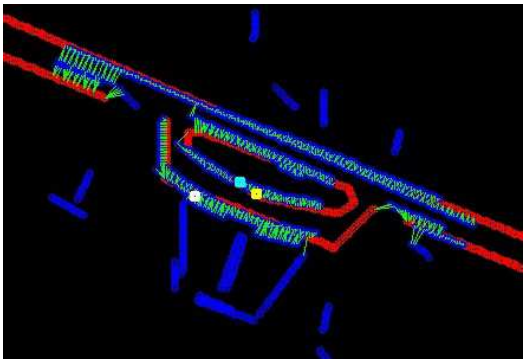
(a) Extracted lines in an UAV image



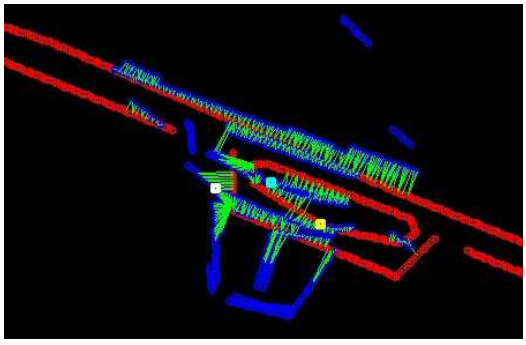
(b) Reference lines from the GIS database



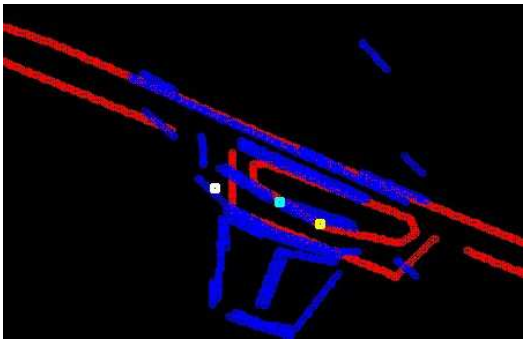
(c) Initial starting point of ICP



(d) Iteration #2 of ICP

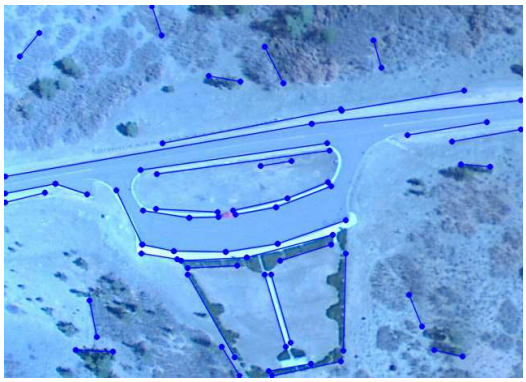


(e) Iteration #20 of ICP

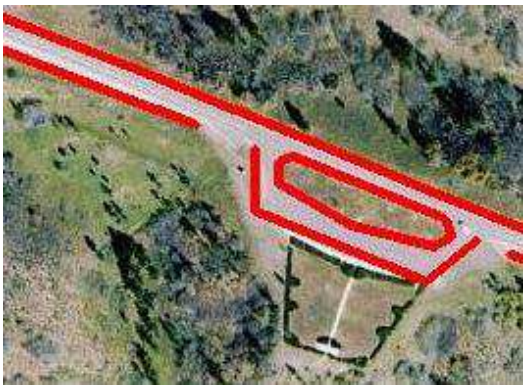


(f) After 45 iterations of ICP

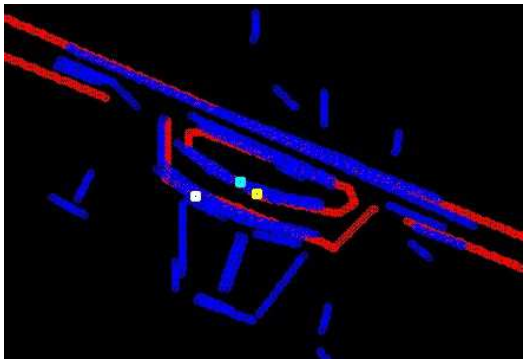
Fig. 7.8. Target localization using ICP. The starting point of ICP was provided by GPS/IMU input. The query lines (blue) approach to the reference lines (red) during the ICP. The corresponding pairs between query and model sets are depicted by green lines. The yellow square indicates the ground truth target location, the white square the target location estimated using the onboard GPS/IMU, and the cyan square the target location estimated by ICP.



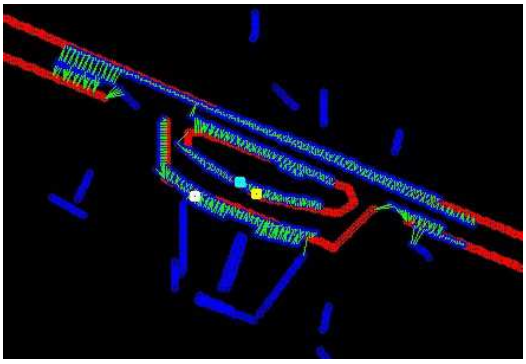
(a) Extracted lines in an UAV image



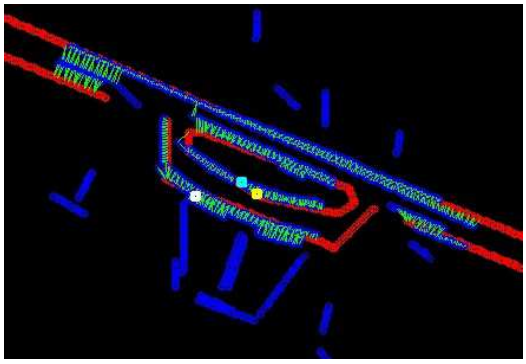
(b) Reference lines from the GIS database



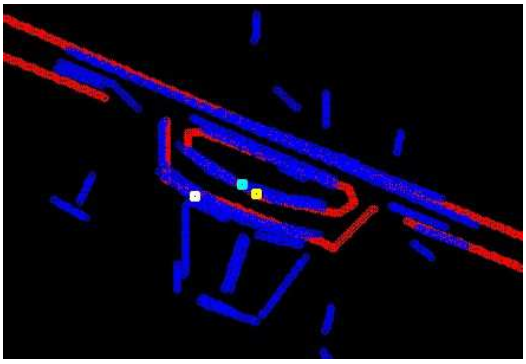
(c) Initial starting point of ICP



(d) Iteration #1 of ICP



(e) Iteration #5 of ICP



(f) After 8 iterations of ICP

Fig. 7.9. Target localization using ICP. The starting point of ICP was provided by the frame-to-frame homography. The query lines (blue) approach to the reference lines (red) during the ICP. The corresponding pairs between query and model sets are depicted by green lines. The yellow square indicates the ground truth target location, the white square the target location estimated using the onboard GPS/IMU, and the cyan square the target location estimated by ICP.

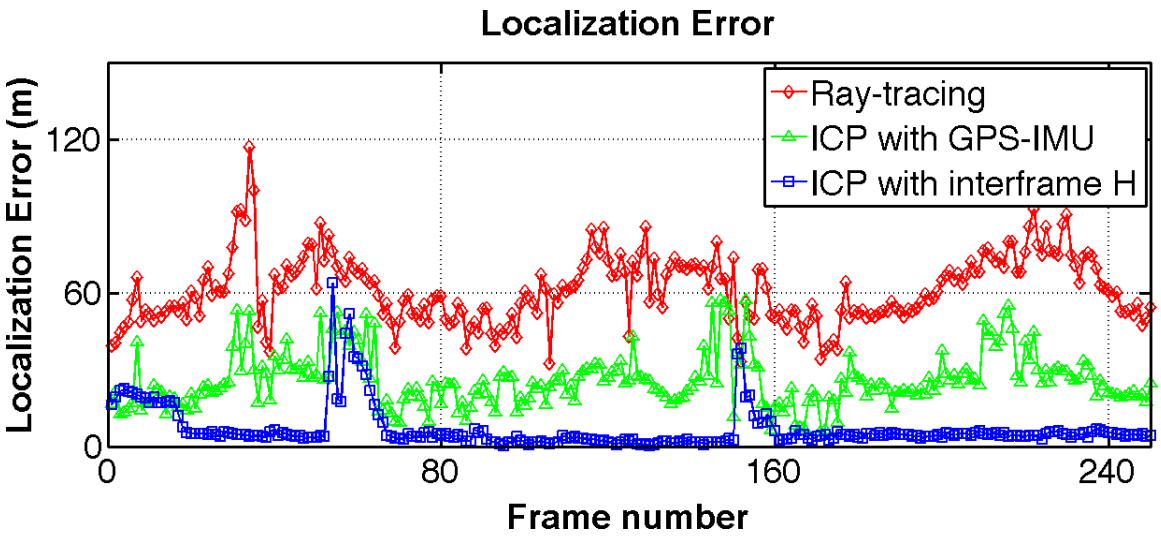


Fig. 7.10. The localization error by three different methods in which ICP was allowed to take an unlimited number of iterations.

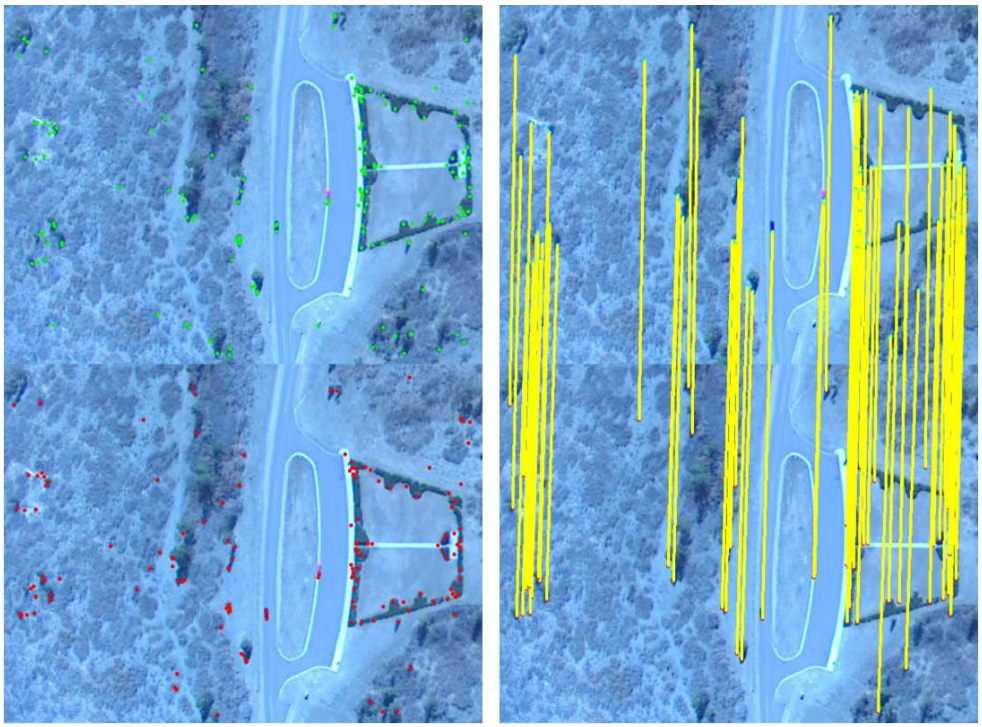


Fig. 7.11. An example of computing homography between two consecutive frames in the image sequence; the green and red dots in the two images in the left column represent the extracted SURF points, and the yellow lines in the right image indicate the matched pairs of SURF points between the two images.

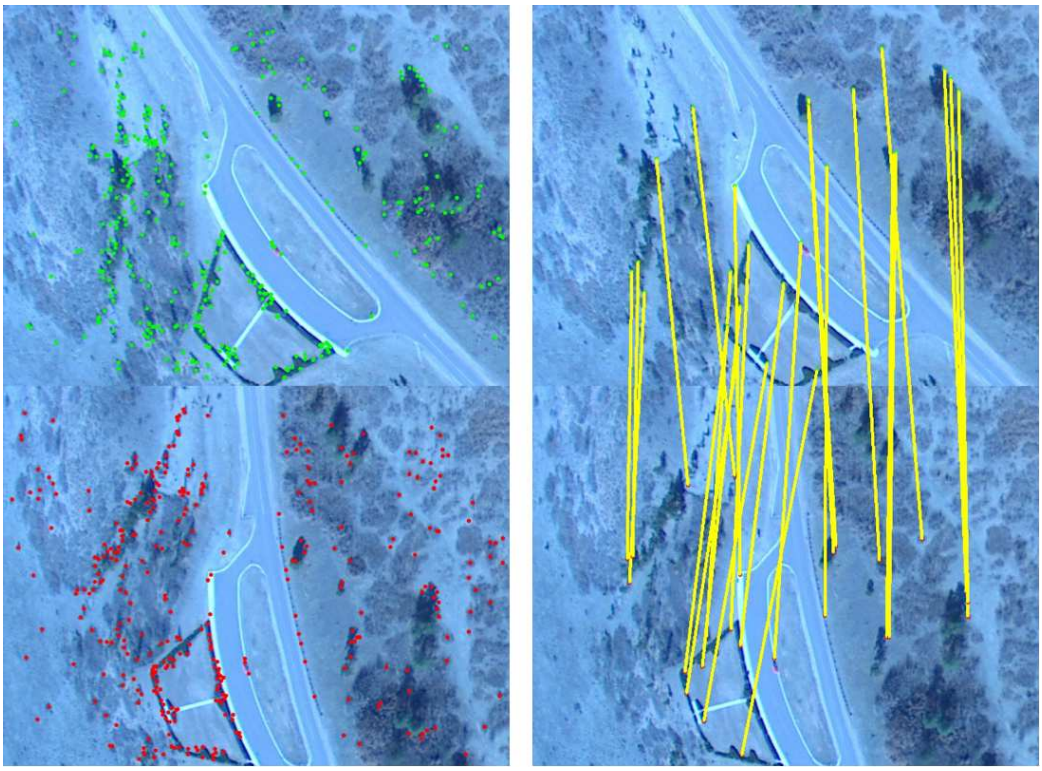


Fig. 7.12. The matched points between two consecutive frames where the resulting homography was rejected because of the large error between the matching points.

tion; (3) the proposed method of ICP with the initial image-to-ground homography computed using interframe homographies. The spikes in the error for the proposed method are associated with frames in which the interframe homography could not be accurately measured, as shown in Fig. 7.12, and thus GPS/IMU initialization was used. The median localization error of our proposed method, 4.24m, was over five times better than ICP with GPS/IMU initialization and over fourteen times better than using ray tracing. Note that when there exist no prominent lines in the current frame, the ICP naturally fail to estimate the correct homography. In the case, we set the initial estimate of camera to ground-to-plane homography $\tilde{\mathrm{H}}_{c_k,w}$ as the $\mathrm{H}_{c_k,w}$.

7.1.6 Time-Bounded ICP

For the results in the previous section, we allowed ICP to iterate until convergence. However, for a real-time system an arbitrary number of iterations is not feasible. We now consider the performance of our proposed method when the number of iterations is limited to a fixed number. Figure 7.13 compares the ground target localization errors for different limits on the number of iterations. The localization error of ICP with an arbitrary number of iterations is depicted by black rectangles in Figure 7.13 and represents the best possible scenario. When we fix the maximum number of iterations, convergence is slowed and it takes a larger number of iterations to reach convergence. However, even with a small number of iterations the error is reduced in each frame and over a large number of frames approaches the best possible performance.

Figure 7.14 shows the error reduction achieved by ICP in each frame when ICP is limited to 15 iterations. Notice that the initial error of each frame is close to the final error of the preceding frame. This is a result of using the interframe homographies to propagate the results from the previous frame into the current frame.

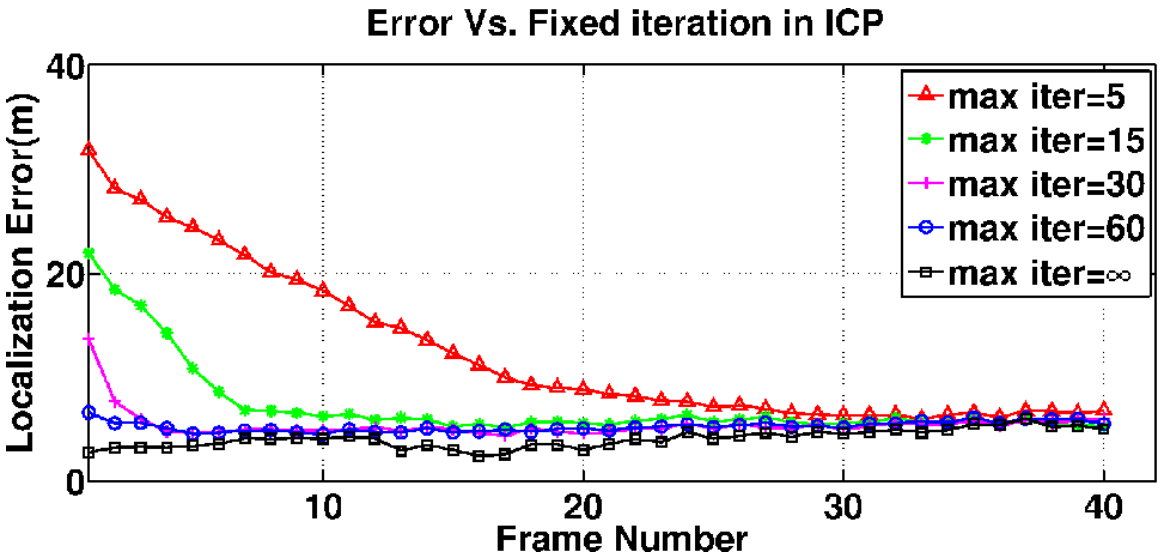


Fig. 7.13. The convergence of target localization error in subsequent inputs with fixed number of iteration in ICP

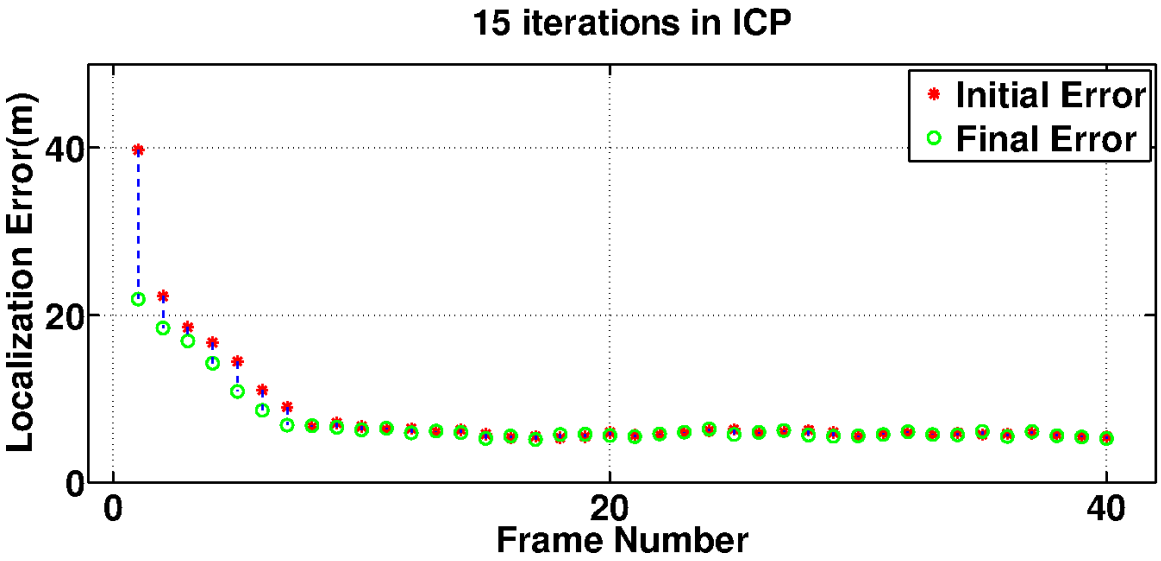


Fig. 7.14. Initial and final error of ICP initialized using interframe homographies.

TABLE 7.2 shows the processing time required for the major components of our algorithm. In our system, new images are acquired at approximately 2fps and so we have limited the maximum number of iterations to 15.

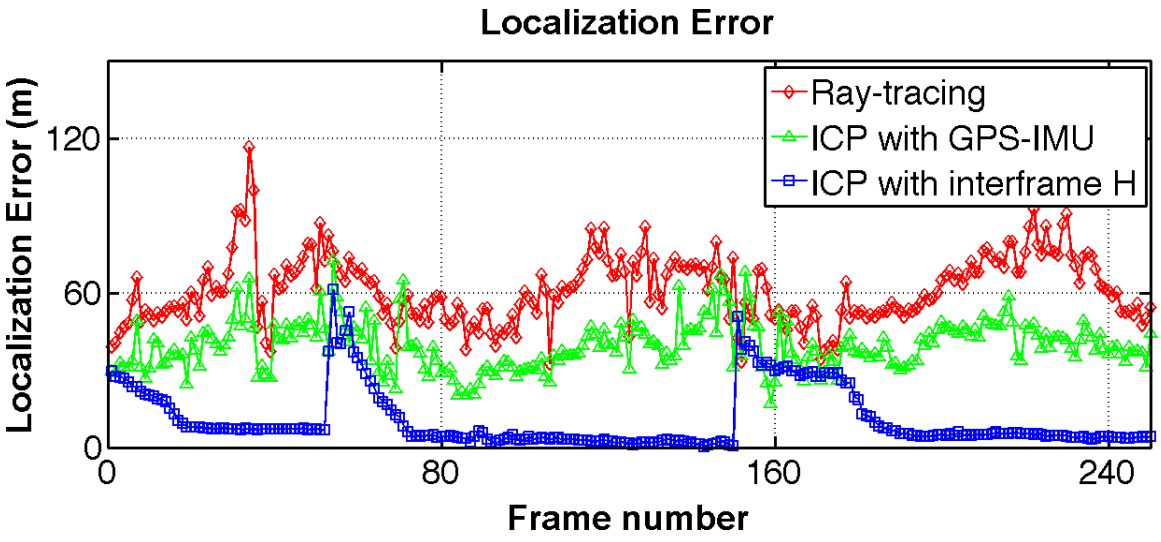


Fig. 7.15. The localization error by three different methods in which ICP was limited to 15 iterations.

Table 7.2.
Average computational requirements of major steps in our algorithm

	Line Extraction	Interframe Homography	ICP iteration
Time (ms)	49.8	0.83	23.6

Table 7.3.
Ground target localization error with ICP limited to 15 iterations

	Ray-Tracing	ICP with GPS/IMU	ICP with interframe
median error(m)	60.2964	39.0408	5.1788

Figure 7.15 shows the ground target localization error for our 250 image dataset while TABLE 7.3 summarizes these results. Notice that the error increases substantially if we initialize ICP using the GPS/IMU but only increases a small amount if we use interframe homographies and previous results. This provides further evidence that our proposed initialization gives good performance even under time constraints.

7.2 Ground target tracking and localization with CMIL

In the previous sections, we showed that the combination of the interframe homography and ICP with line matching can estimate the image-to-ground homography of the current frame. The interframe homography initialized the image-to-ground homography of the current frame and ICP updated this initial image-to-ground homography. The image-to-ground homography can localize the ground target in well-structured environments where there exist a number of man-made structures that can provide sufficient lines for ICP.

In this section, we concentrate on how to track and localize the moving ground target, the human subject, in unstructured environments. Under unstructured environments where there exist few man-made structures, ICP could fail to update the initial image-to-ground homography due to an insufficient number of lines that mainly come from man-made structures. The failure of ICP yields the updated image-to-ground homography as being identical to the initial image-to-ground homography; it deteriorates the ground target localization in the ground coordinate space due to the inaccurate image-to-ground homography. Therefore, we directly estimate the ground target location by tracking the ground target in the image coordinate space.

One hurdle for tracking the ground target, especially the human on the ground, is the fact that the shape and appearance of the moving human on the ground often varies with respect to changes in the viewing angle of cameras attached to the UAVs. The different viewing angles in each frame are usually caused by different moving speeds between UAVs and humans or abrupt changes of the moving path of UAVs from unexpected wind turbulence, control malfunction, and so on. An additional factor of shape and appearance changes in the human on the ground is to interact with ground objects, e.g, opening a car door, entering a building, walking beside cars, etc. Consequently, the severe changes of shape and appearance of the human on the ground make it hard to find the best estimated position of the human subject, even with simple motions. Another problem of human tracking for UAV vision is the fact that both human motions and UAV motions simultaneously contribute to observed human movements in the image coordinate space. Therefore, it is required to manipulate interconnection between human motions and UAV motions.

We apply the CMIL with a particle filtering framework for tracking humans on the ground with severe shape and appearance changes due to either viewing angle variations or interactions with other ground objects. The CMIL framework can first address problems of tracking human subjects with severe changes of shape and appearance through updating the reference template of the human subject. The trained classifiers in each frame can serve to update the reference template for the human

subject. Interframe homography works as the motion prediction model for CMIL with a particle filter framework. Interframe homography implicitly deals with both human motions and UAV motions in the image coordinate space and also can be adaptively changed during tracking. Therefore, we expect that CMIL with particle filtering embedding interframe homography can provide a good human tracking algorithm with UAV vision.

7.2.1 Motion Prediction Model with Interframe Homography

The typical non-linear motion prediction model in particle filtering is expressed by

$$\mathrm{X}_t = f(\mathrm{X}_{t-1}, \mathrm{V}_t), \tag{7.16}$$

where X_t is the state of the particle at time t and V_t is the noise at time t.

Given a set of particles $\mathbf{x}_t$, the motion prediction in the particle filtering is conventionally used to sample new particles $\tilde{\mathbf{x}}_{t+1}$ through the transition density,

$$\tilde{\mathbf{x}}_{t+1} \sim p(\mathbf{x}_{t+1}|\mathbf{x}_t). \tag{7.17}$$

For tracking human subjects from UAV images, it is difficult to derive a correct motion prediction model because the motion prediction model for human subjects should simultaneously consider the human motions as well as UAV motions. Instead of finding the correct motion prediction model, we can replace the motion prediction model with interframe homography. Interframe homography can connect one point in the first UAV image with the corresponding point in the second UAV image; two UAV images are used to estimate interframe homography. With interframe homography H_{c_{t+1},c_t}, the set of particle $\mathbf{x}_t$ is transferred into the corresponding set of particle $\mathbf{x}'_{t+1}$,

$$\mathbf{x}'_{t+1} = \mathrm{H}_{c_{t+1},c_t}\mathbf{x}_t. \tag{7.18}$$

The motion prediction model in Eq. 7.17 can be converted as

$$\tilde{\mathbf{x}}_{t+1} \sim p(\mathbf{x}_{t+1}|\mathbf{x}'_{t+1}). \tag{7.19}$$

Fig. 7.16 illustrates the motion prediction model combined with interframe homography for CMIL particle filtering. For graphical clarification, the particles in the frame at time t are assumed to be uniformly distributed. The top row shows the conventional motion prediction model, and the bottom row displays the motion prediction model combined with the interframe homography in particle filtering. The main difference between them is the initial position of particles at time $t+1$. The initial positions of particles at time $t+1$ are the same positions of particles $\mathbf{x}_t$ in the conventional motion prediction model. However, the positions of particles $\mathbf{x}_t$ are transferred into the positions of the particles at $t+1$ through interframe homography H_{c_{t+1},c_t} as shown in bottom row of Fig. 7.16. Therefore, we expect that the initial positions of particles at $t+1$ with the interframe homography can formalize a better candidate motion prediction model.

We utilize two distance measures for CMIL with particle filtering. These two distance measures, d_B and d_{max}, are used for computing the weight of each particle. Let $\boldsymbol{\nu} = \left\{\nu^{(\ell)}|\ell = 1, \cdots, L\right\}$ be a set of confidence scores where $\nu^{(\ell)}$ is the output of the classifier $\mathbf{C}_t$ applied to the ℓ-th component in the image patch. The feature vectors of an image patch are composed of two parts: (1) the histogram of the confidence score set $\boldsymbol{\nu}$ and (2) the maximum value in the confidence score set. Denoting the histogram as $\mathbf{r}$ and $\nu_{max} \equiv \max(\boldsymbol{\nu})$, the feature vector of an image patch at $\mathbf{x}_t^{(i)}$ is defined as:

$$\mathbf{s}(\mathbf{x}_t^{(i)}) = [\mathbf{r}(\mathbf{x}_t^{(i)}), \nu_{max}(\mathbf{x}_t^{(i)})]. \tag{7.20}$$

We use the Bhattacharyya distance [54] as the first distance measure d_B for comparing the two histograms. The Bhattacharyya distance is defined as :

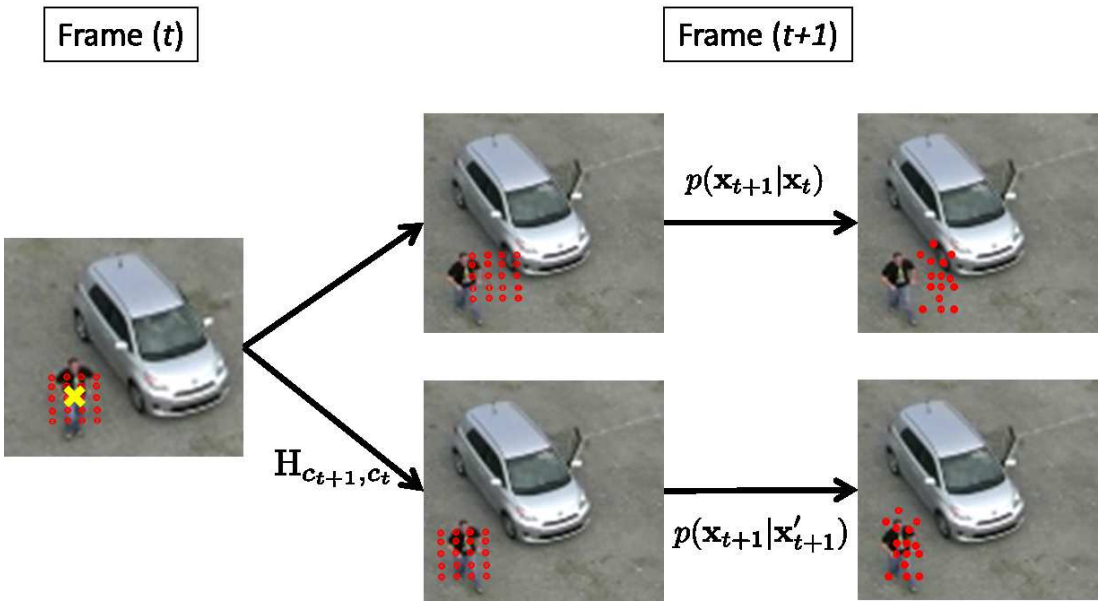


Fig. 7.16. Comparison between motion prediction in conventional particle filtering and that in particle filtering with interframe homography. The red dots represent particles

$$d_B(p,q) = \sqrt{1 - \sum_{i=1}^{N} \sqrt{p(i)q(i)}}, \tag{7.21}$$

where p and q are discrete distributions. As demonstrated in [44], the maximum confidence score in a positive bag in MIL can be used for estimating the probability of the positive bag as

$$\Pr(y = 1|\nu) \propto \frac{1}{e^{-\nu_{max}}}, \tag{7.22}$$

where y is the label of bag, $y \in \{-1, 1\}$. The second distance measure d_{max} is defined as

$$d_{max}(\nu_{max}^{(k)}, \nu_{max}^{(\ell)}) = \nu_{max}^{(k)} - \nu_{max}^{(\ell)}, \tag{7.23}$$

where $\nu_{max}^{(k)} = \nu_{max}(\mathbf{x}^{(k)})$.

Finally, the weight of each particle is then given by

$$\begin{array}{rcl} w_{t+1}^{(i)} & \approx & p(\mathbf{z}_{t+1}|\mathbf{x}_{t+1}^{(i)}) \\ & \propto & e^{-\gamma(\alpha d_B(\mathbf{r}(\hat{\mathbf{x}}_t), \mathbf{r}(\mathbf{x}_{t+1}^{(i)})) + (1-\alpha) d_{max}(\hat{\nu}_t, \nu_{t+1}^{(i)}))} \end{array}, \tag{7.24}$$

where $\mathbf{z}_{t+1}$ is an observed feature vector at $\mathbf{x}_{t+1}^{(i)}$, $\hat{\nu}_t = \nu_{max}(\hat{\mathbf{x}}_t)$, $\nu_{t+1}^{(i)} = \nu_{max}(\mathbf{x}_{t+1}^{(i)})$, and γ and α are user-specified control parameters.

7.2.2 Experiments

We tested the CMIL with particle filtering for tracking the human subject on test sequences which have been taken in unstructured environments, as shown in Figure 7.17: *action* 1 and *action* 2 sequences[1]. Test sequences, which are composed of a series of human movements, such as opening a car door and walking out of the car were captured from a 400-to-450-foot flying height with different viewing angles. The ground coordinates in Figure 7.17 are measured by using Google Earth and are set

[1]http://crcv.ucf.edu/data/UCF_Aerial_Action.php

with the x axis to the east direction and the y axis to the south direction. As was the case in comparing in order to localize the stationary ground target in structured environments in the previous section, this work focused on the localization of the moving ground target in unstructured environments.

We compare four different tracking algorithms: (1) MIL tracking (MIL), (2) MIL tracking with interframe homography (MIL-IH), (3) CMIL with particle filtering (CMIL-PF), and (4) CMIL with particle filtering through interframe homographies (CMIL-PFIH). Among them, MIL-IH, which is newly introduced to compare the proposed CMIL-PFIH, transfers the current target position into an initial search position in the next frame with interframe homography. Some parameters used in particle filtering are empirically selected; we have fixed the number of particles to be 70 and the noise standard deviation of state transition to be 8 pixels in all test sequences. We have chosen the parameters for computing the weight of each particle to be $\gamma = 3$ and $\alpha = 0.3$ in CMIL-PF and CMIL-PFIH trackers. Similar to previous experiments in CMIL-based tracking, we have analyzed both qualitative and quantitative performance of trackers.

Figure 7.18 and Figure 7.19 show the qualitative assessment of the trackers by displaying the bounding box at the position of the human target estimated by each tracker in each frame. The center position of each rectangle is the best estimated position of the human target in the image coordinate space. The *action* 1 sequence in Figure 7.18 consists of following human actions: opening the car door, getting out of the car, closing the car door, and walking away from the car. CMIL-PFIH, depicted by blue rectangles, showed the best tracking results among four tracking algorithms. It successfully encloses the human subject on different human motions. Figure 7.19 displays tracking results on the *action* 2 sequence that has 2 different humans; one is walking away from the left side of the car while another gets out of the car. Similar to results depicted in Figure 7.18, CMIL-PFIH outperforms other tracking algorithms for tracking a human whose motion is to walk away from the left side of the car.

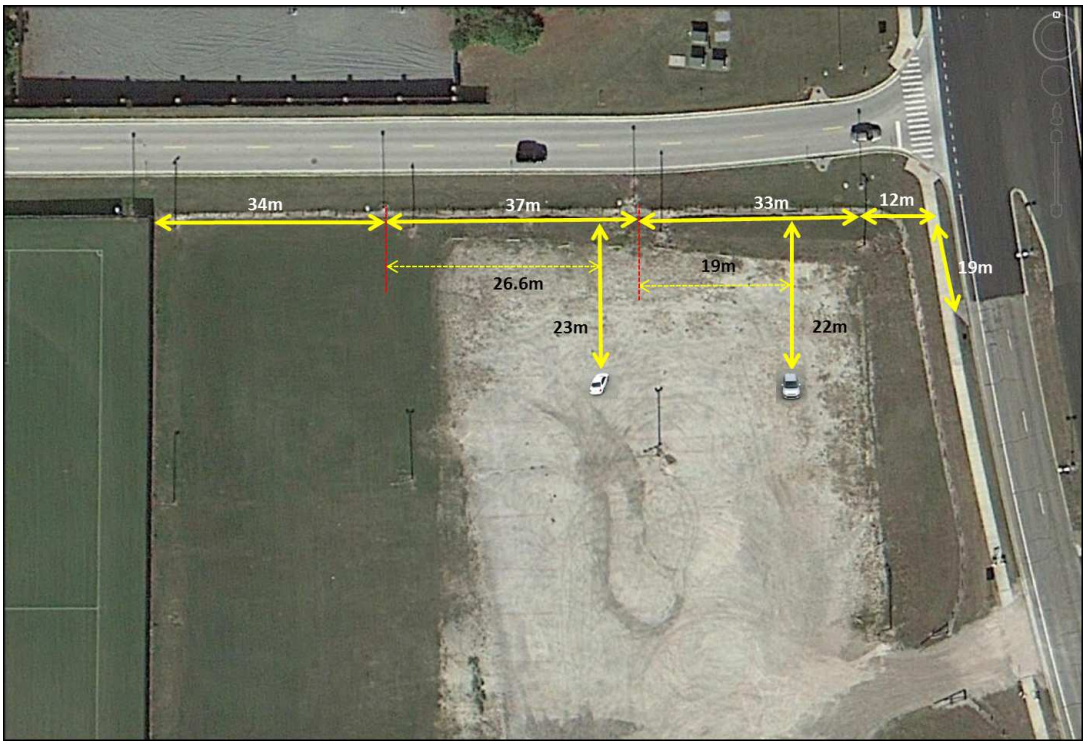


Fig. 7.17. The test site for capturing action 1 and action 2 video sequences. The distance between locations in the image are measured by Google Earth's distance measuring tool. This site is located in the north area of the University of Central Florida campus.

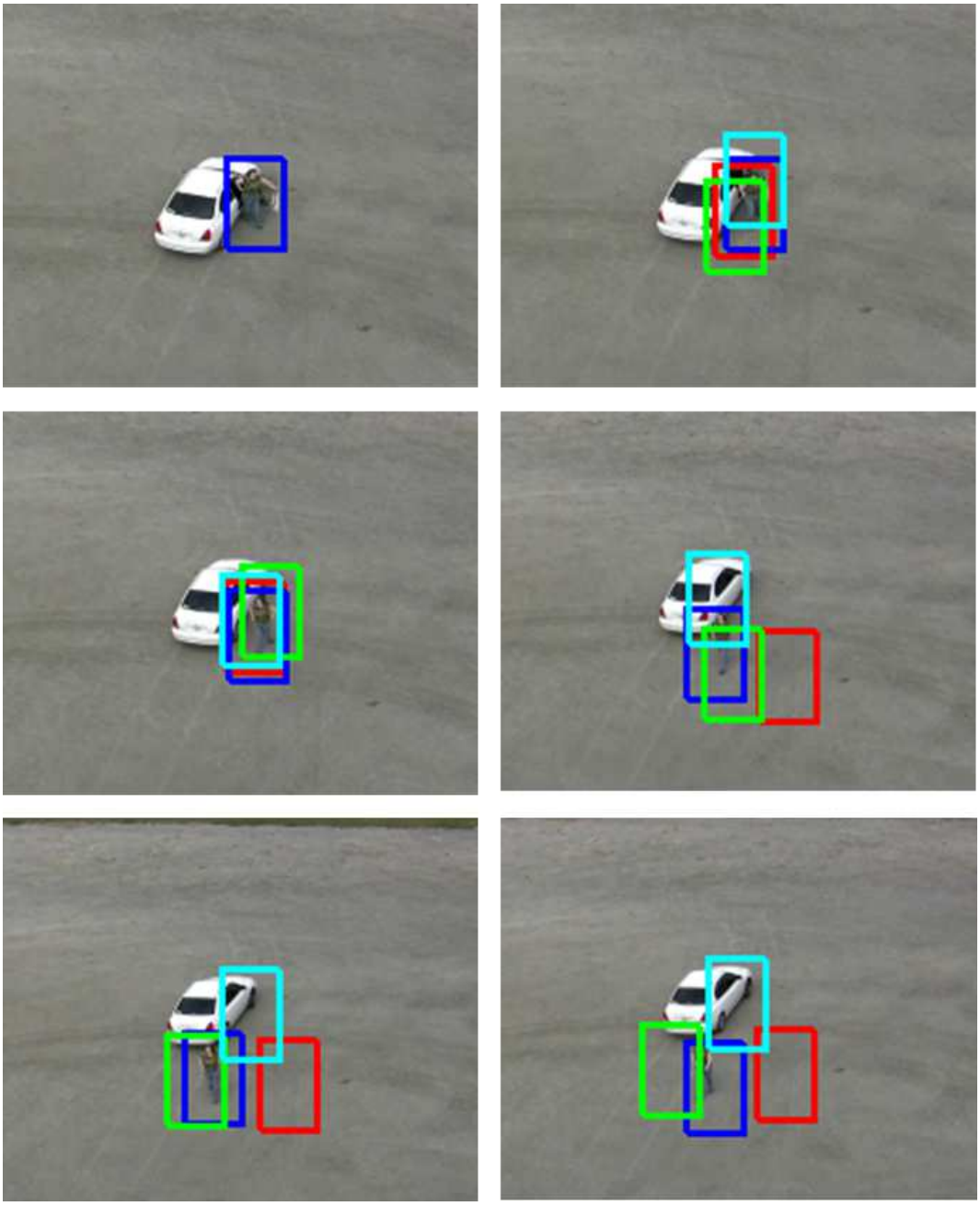


Fig. 7.18. Example of the tracking results of action 1 video sequence. The four trackers compared are : (1) MIL (red rectangles), (2) CMIL-PF (green rectangles), (3) MIL-IH (cyan rectangles), and CMIL-PFIH (blue rectangles)

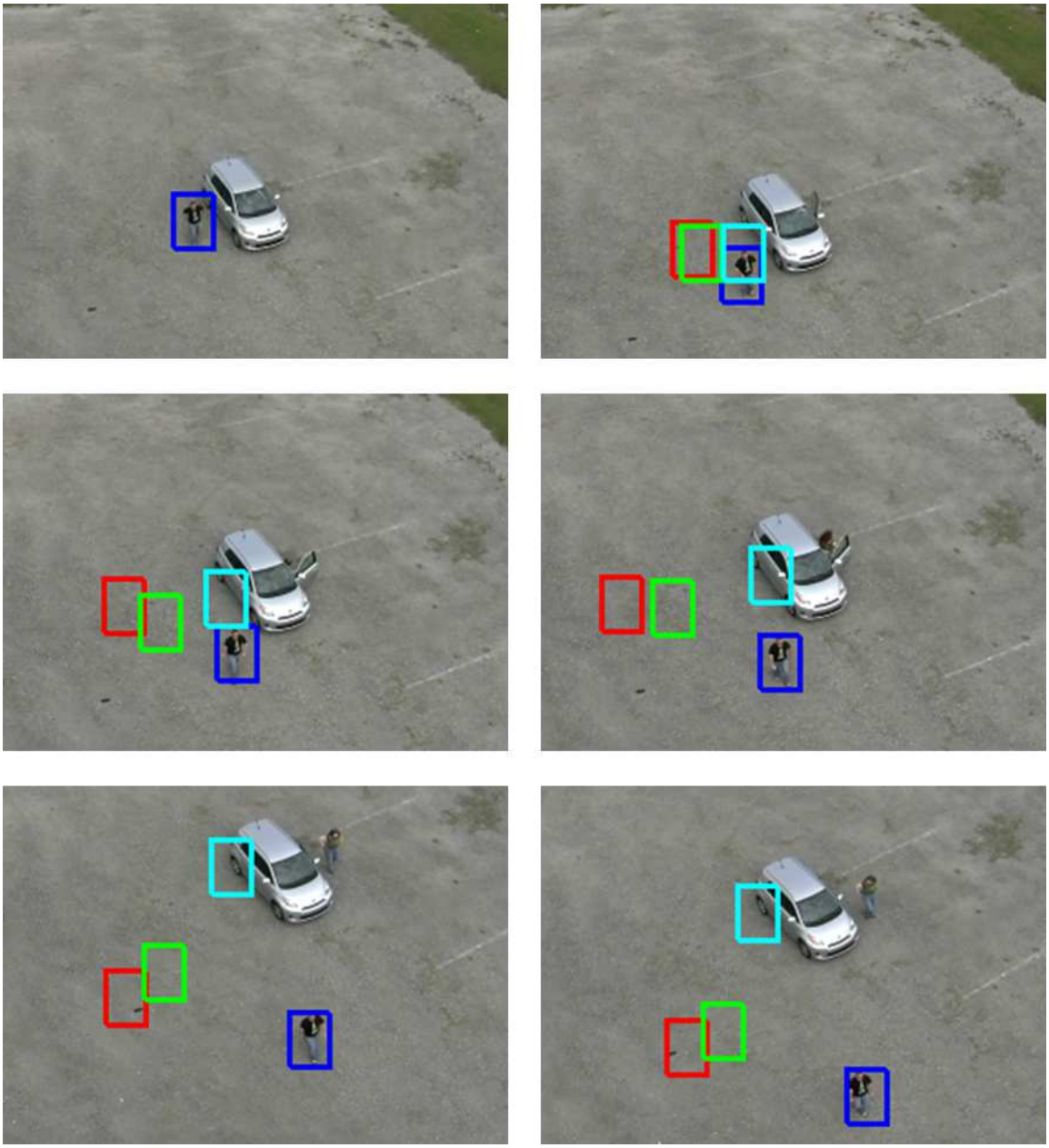


Fig. 7.19. Example of the tracking results of action 2 video sequence. The four trackers compared are : (1) MIL (red rectangles), (2) CMIL-PF (green rectangles), (3) MIL-IH (cyan rectangles), and CMIL-PFIH (blue rectangles)

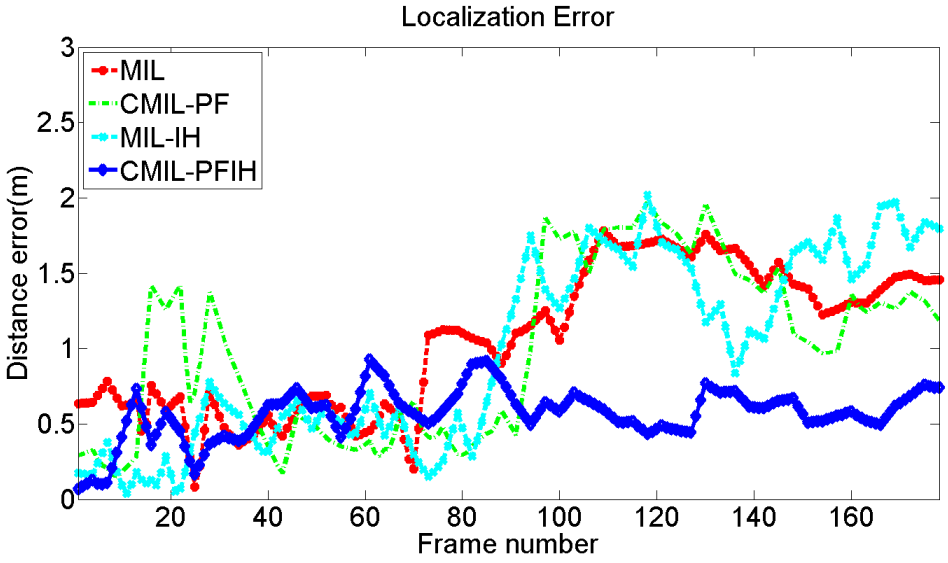


Fig. 7.20. Localization error on *action* 1 video sequence

The quantitative assessment of the trackers are carried out by localization error in the ground coordinate space. Both the center position of each rectangle and the ground truth position of the human subject in the image coordinate space are transformed into corresponding points in the ground coordinate space through the image-to-ground homographies, respectively. Figure 7.20 and Figure 7.21 display the localization errors of the four compared trackers in *action* 1 and *action* 2 test sequences, respectively. CMIL-PFIH tracker shows the best tracking performance in both test sequences. On the other hand, three trackers, MIL, CMIL-PF, and MIL-IH showed large localization errors in the *action* 2 sequence as shown in Figure 7.21. Table 7.4 summarizes the average localization errors of four trackers in the ground coordinate space.

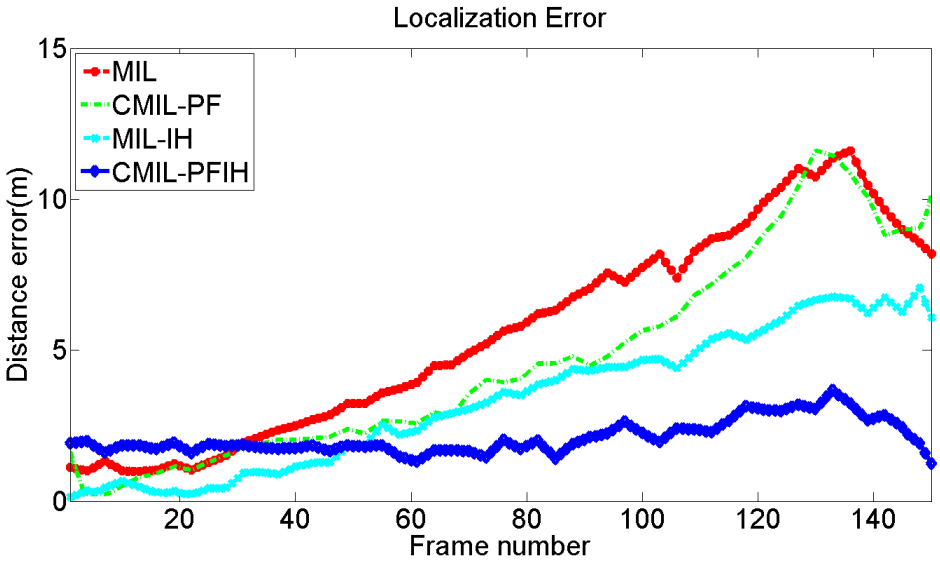


Fig. 7.21. Localization error on *action* 2 video sequence

Table 7.4.
Mean localization error in four trackers

	MIL	CMIL-PF	MIL-IH	CMIL-PFIH
action 1	1.06	1.01	0.98	0.57
action 2	4.66	5.58	3.33	2.11

7.3 Conclusion

We presented a new method for improving the accuracy of ground target localization. We utilized line features to match the UAV images to the GIS database by ICP algorithm. We used line features to efficiently represent structured environment in a GIS database. In order to address time constraints and to increase the likelihood of convergence to the correct ground target localization, we propagated the image-to-ground homography across successive frames using interframe homographies. This propagated image-to-ground homography efficiently initialized ICP for the current frame. We demonstrated that the propagated image-to-ground homographies increased the accuracy of ground target localization in the experiments. The experimental results showed that this initialization improved the accuracy of ground target localization after a few frames even when limiting the number of iterations in ICP.

Along with this, we demonstrated the combination CMIL-PF with interframe homography for moving ground target localization. Interframe homography worked as the motion prediction model for CMIL-PF. The motion prediction model with the interframe homography efficiently compensated for ground target motions triggered by both target movements and UAV movements. The experimental results displayed that the CMIL-PFIH outperforms other comparing tracking algorithms even in unstructured environments.

8. SUMMARY

8.1 Conclusions

In this thesis, we explored the problems of articulated human movement tracking utilizing by the tracking-by-detection framework. We adapted the instance-learning-based classifier for making a decision about the location of the human target in video sequences. Specifically, we developed the enhanced instance learning-based classifier, which is based on the multiple instance learning classifier, since articulated human movements are clearly non-rigid motions.

We first developed the new basic classification unit of MIL so that the classifier can resolve the large variation of human movements in successive frames. The basic classification unit, *component*, is obtained by automatically segmenting both positive and negative image patches. Components provide our approach with extra degrees of freedom with which the tracker can deal with the large frame-to-frame variations caused by articulated human movements. CMIL tracking was also combined with the online boosting learning system for updating the classifier that can detect the human target to be tracked subsequently.

We then extended the basic CMIL approach by analyzing sample distributions for the training stage. The positive image patch in this extension was classified by patches that have more than the minimum number of positive components. The minimum number of positive components was updated in the online version by analyzing the distribution of positive and negative components in positive image patches. The extended CMIL approach with the adaptively configured positive image patches outperformed other tracking algorithm as compared in experiments.

The probabilistic approach has augmented the basic CMIL tracking by combining with the motion prediction framework. The probabilistic prediction has resolved the

issue arising from the large and rapid translational human movements which easily break down the in-place articulated movement tracking. The positive image patches in CMIL worked as the motion prediction model in the particle filter and provided more accurate likelihood estimation. With the extension of this motion prediction framework, we achieved the competitive tracking performance for large and rapid human transnational movements and further evaluated the tracking performance for ground objects for unmanned aerial vehicles vision.

Finally the comparative study with the state-of-the-art CNN based tracking suggested that the proposed CMIL trackings are competitive to CNN based tracking for articulated human movements. Additionally, CMIL-PF sufficiently performed in the sequence including perspective distortion from the top-view angle.

8.2 Future Works

From the viewpoint of unifying two main human tracking approaches, in this thesis we have sought to integrate both the appearance-based tracking and part-based tracking for articulated human movement tracking. The appearance-based approach can be described by the unique pixel distribution in the component, and the part-based approach can be expressed by the segmentation as well as the interconnection between components, which is modeled by the set of weak classifiers. We need to take into account the following four issues for improving articulated human movement tracking performance.

First, we need to utilize more information for extracting components from image patches. In Chapter 3, grayscale-based segmentation has been used for extracting components in each image patch. The color-based segmentation can be useful to discriminate the human and the background in some environments, and the specific image features can also feasibly set up the initial segmenting seeds for complex image contents. However, we also need to consider a trade-off between computational speed and segmenting performance.

The second issue is which learning technique has to be selected to enhance the current component-based learning scheme. In this thesis, we applied the boosting learning technique due to the marginal convergence of its online version with the offline version. However, as described in Chapter 2, the online boosting can only select a specific weak classifier from the pool that contains a finite number of the weak classifier. Therefore the final strong classifier has the limited tracking performance due to its finite possible configurations. The recent progress on computation power may enable the application of other online learning techniques, such as online SVM, online convex regularization, etc.

The third issue is how to define the positive image patch in the context of MIL. The original definition of the positive image patch in MIL is the patch that contains at least one positive component. This may cause more false-positive image patches that have been inappropriate for positive learning samples. In Chapter 4, the definition of positive image patch is extended into the patch that contains at least the specified minimum number of positive components and the required minimum number of positive components can be adaptively varied with the distribution of positive image patches. We need to enhance the way of defining the minimum number of positive components.

The final issue is how to incorporate the motion prediction framework into the current CMIL tracking framework. In Chapter 5 and 7, we integrated the particle filter with the component based MIL as a candidate framework and evaluated it for articulated human movement tracking in different environments. However, the problem is how to set up a suitable motion prediction model for covering the large articulated human movements, which can be represented by the state transition probability $p(x_t \mid x_{t-1})$ in the particle filter framework. The human kinetic model can be a candidate transition model for motion prediction.

REFERENCES

REFERENCES

[1] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 34, no. 4, pp. 743–781, 2012.

[2] M. Bertozzi, A. Broggi, A. Fascioli, and A. Tibaldi, "Pedestrian localization and tracking system with kalman filtering," 2004, pp. 584–589.

[3] G. Grubb, A. Zelinsky, L. Nilsson, and M. Rilbe, "3d vision sensing for improved pedestrian safety," in Intelligent Vehicles Symposium, 2004 IEEE, june 2004, pp. 19 – 24.

[4] J. Giebel, D. M. Gavrila, and C. Schnorr, "A bayesian framework for multi-cue 3d object tracking," in In Proceedings of European Conference on Computer Vision, 2004, pp. 241–252.

[5] K. Wnuk and S. Soatto, "Multiple instance filtering," in Proc. of NIPS, vol. 65, 2011, p. 71.

[6] C. Aeschliman, J. Park, and A. C. Kak, "A probabilistic framework for joint segmentation and tracking," in Computer Vision and Pattern Recognition, 2010. CVPR 2010. IEEE Conference on, 2010.

[7] Y. Li, Z. Zhou, and W. Wu, "Iterative pedestrian segmentation and pose tracking under a probabilistic framework," in IEEE International Conference on Robotics and Automation, 2012.

[8] S. Gammeter, A. Ess, T. Jaggli, K. Schindler, B. Leibe, and L. V. Gool, "Articulated multi-body tracking under egomotion," in European Conference on Computer vision, 2008.

[9] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, "Survey of pedestrian detection for advanced drive assistance systems," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 32, pp. 1239–1258, 2010.

[10] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," in Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008, pp. 1–8.

[11] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool, "Online multiperson tracking-by-detection form a single, uncalibrated camera," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 33, no. 9, pp. 1820–1833, 2011.

[12] Z. Lin, G. Hua, and L. Davis, "Multiple instance feature for robust part-based object detection," in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009, pp. 405–412.

[13] L. Zhang, B. Wu, and R. Nevatia, "Detection and tracking of multiple humans with extensive pose articulation," in Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. IEEE, 2007, pp. 1–8.

[14] B. Yang, C. Huang, and R. Nevatia, "Extensive articulated human detection by voting cluster boosted tree," in Applications of computer vision 2009 Workshop on. IEEE, 2009, pp. 1–8.

[15] N. X. Tuong, T. Muller, and A. Knoll, "Robust pedestrian detection and tracking from a moving vehicle," pp. 78 780H–78 780H–12, 2011. [Online]. Available: + http://dx.doi.org/10.1117/12.871994

[16] M. Soga, T. Kato, M. Ohta, and Y. Ninomiya, "Pedestrian detection with stereo vision," in Data Engineering Workshops, 2005. 21st International Conference on, april 2005, p. 1200.

[17] S. Shen, M. Tong, H. Deng, Y. Liu, X. Wu, K. Wakabayashi, and H. Koike, "Model based human motion tracking using probability evolutionary algorithm," Pattern Recognition Letters, vol. 29, no. 13, pp. 1877 – 1886, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016786550800202X

[18] P. Dollar, B. Babenko, S. Belongie, P. Perona, and Z. Tu, "Multiple component learning for object detection," in European Conference on Computer vision, 2008.

[19] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 32, no. 9, pp. 1617–1645, 2009.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," International Journal of Computer Vision (IJCV), vol. 115, no. 3, pp. 211–252, 2015.

[21] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural network," in Computer Vision and Pattern Recognition, 2014. CVPR 2014. IEEE Conference on. IEEE, 2014.

[22] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in Computer Vision, 2016. ECCV 2016. 14th European Conference on. Springer, 2016.

[23] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in Computer Vision and Pattern Recognition, 2017. CVPR 2017. IEEE Conference on. IEEE, 2017.

[24] A. Bulat and G. Tzimiropoulos, "Human pose estimation via convolutional part heatmap regrgression," in Computer Vision, 2016. ECCV 2016. 14th European Conference on. Springer, 2016.

[25] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy, "Toward accurate multi-person pose estimation in the wild," in Computer Vision and Pattern Recognition, 2017. CVPR 2015. IEEE Conference on. IEEE, 2017.

[26] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machine," in Computer Vision and Pattern Recognition, 2016. CVPR 2016. IEEE Conference on. IEEE, 2016.

[27] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in Computer Vision, 2017. ICCV 2017. IEEE Conference on. IEEE, 2017.

[28] R. Girshick, "Fast r-cnn," in Computer Vision, 2015. ICCV 2015. IEEE Conference on. IEEE, 2015.

[29] R. A. Guler, N. Neverova, and I. Kokkinos, "Densepose: Dense human pose estimation in the wild," in Computer Vision and Pattern Recognition, 2018. CVPR 2018. IEEE Conference on. IEEE, 2018.

[30] R. A. Güler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, and I. Kokkinos, "Densereg: Fully convolutional dense shape regression in-the-wild," in Computer Vision and Pattern Recognition, 2017. CVPR 2017. IEEE Conference on. IEEE, 2017.

[31] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. Zitinick, "Microsoft coco: Common objects in context," in Computer Vision, 2014. ECCV 2014. 12th European Conference on. Springer, 2014.

[32] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, "Solving the multiple-instance problem with axis-paralle rectangles," Artificial Intelligence, vol. 89, pp. 31–71, 1997.

[33] O. Maron and T. Lozano-Perez, "A framework for multiple-instance learning," in Advances in Neural Information Processing Systems. MIT Press, 1998, pp. 570–576.

[34] Y. Chen, J. Bi, and J. Z. Wang, "Miles: Multiple-instance learning via embedded instance selection," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 1931–1947, 2006.

[35] Z. Fu, A. Robles-Kelly, and J. Zhou, "Milis: Multiple instancle learning with instance selection," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 958–977, 2011.

[36] P. Sharma, C. Huang, and R. Nevatia, "Unsupervised incremental learning for improved object detection in a video," in International Conference on Computer Vision and Pattern Recognition, 2012.

[37] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 33, no. 8, pp. 1619–1632, 2010.

[38] P. Viola, J. C. Platt, and C. Zhang, "Multiple instance boosting for object detection," in Proc. Neural Information Processing System, 2005, pp. 1417–1426.

[39] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in Proc. Conf. British Machine Vision, 2006, pp. 47–56.

[40] Y. Freund and R. Schapire, "A decision-theoritic generalization of on-line learning and an application to boosting," Journal of Computer and System Science, vol. 55, pp. 119–139, 1997.

[41] N. C. Oza and S. Russell, "Online bagging and boosting," in In Artificial Intelligence and Statistics 2001, 2001.

[42] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, pp. 2290–2297, 2009.

[43] W. Li and N. Vasconcelos, "Multiple instance learning for soft bags via top instances," in Computer Vision and Pattern Recognition, 2015. CVPR 2015. IEEE Conference on. IEEE, 2015.

[44] M. Kim and F. De la Torre, "Gaussian processes multiple-instance learning," in International Conference on Machine Learning, 2010.

[45] K. Han, J. Park, and A. C. Kak, "Tracking articulated human movements with a component based approach to boosted multiple instance learning," in IEEE International Conference on Image Processsing, 2013.

[46] ——, "Robust tracking of articulated human movements through component-based multiple instance learning with particle filtering," in Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on, 2014.

[47] M. W. Lee, I. Cohen, and S. K. Jung, "Particle filter with analytical inference for human body tracking," in IEEE Workshop on Motion and Video Computing, 2002.

[48] M.Isard and J. MacCormick, "Bramble: A bayesian multiple-blob tracker," in International Conference on Computer Vision, 2001.

[49] K. Okuma, A. Taleghani, N. D. Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking," in the European Conference on Computer Vision, 2004.

[50] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, "Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans," IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1728–1740, 2008.

[51] C. Song, J. Son, S. Kwak, and B. Han, "Dynamic resource allocation by ranking svm for particle filter tracking," in British Machine Vision Conference, 2011.

[52] Z. Ni, S. Sunderrajan, A. Rahimi, and B. Manjunath, "Particle filter tracking with online mulitple instance learning," in Internation Conference on Pattern Recognition, 2010.

[53] Y. Song and Q. Li, "Visual tracking based on mulitple instance learning particle filter," in International Conference on Mechatronics and Automation, 2011.

[54] K. Nummiaro, E. Koller-Meier, and L. V. Gool, "An adaptive color-based particle filter," Image and Vision Computing, vol. 21, pp. 99–110, 2002, bhattacharyya coefficient and distance.

[55] Y.Song, C.Ma, L.Gong, J.Zhang, and a. M. R.Lau, "Crest: Convolutional residual learning for visual tracking," in Computer Vision and Pattern Recognition, 2017. CVPR 2017. IEEE Conference on. IEEE, 2017.

[56] L. Wang, W. Ouyang, X. Wang, and H. Lug, "Visual tracking with fully convolutional network," in Computer Vision, 2015. ICCV 2015. IEEE Internation Conference on. IEEE, 2015.

[57] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in Computer Vision and Pattern Recognition, 2016. CVPR 2016. IEEE Conference on. IEEE, 2016.

[58] D. Gordon, A. Farhadi, and D. Fox, "Re3 : Real-time recurrent regression networks for object tracking," in arXiv preprint arXiv:1705.06368. arXiv, 2017.

[59] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in In European Conference on Computer Vision. Springer, 2016.

[60] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in Computer Vision and Pattern Recognition, 2017. CVPR 2017. IEEE Conference on. IEEE, 2017.

[61] B. Li, J. Yan, W. Wu, S. Shu, and X. Hu, "High performance visual tracking with siamese region proposal network," in Computer Vision and Pattern Recognition, 2018. CVPR 2018. IEEE Conference on. IEEE, 2018.

[62] H. Nam and B. Han, "Learning multi-domain convolutional neural network for visual tracking," in Computer Vision and Pattern Recognition, 2016. CVPR 2016. IEEE Conference on. IEEE, 2016.

[63] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in Computer Vision, 2016. ECCV 2016. 14th European Conference on. Springer, 2016.

[64] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in Image Processing, 2017. ICIP 2017. IEEE Conference on. IEEE, 2017.

[65] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Learning Representations, 2015. International Conference on. IEEE, 2015.

[66] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L.Cehovin, G.Nebehay, T.Vojır, G.Fernandez, and et al., "The visual object tracking vot2014 challenge results," in Computer Vision, 2014. ECCV 2014. 12th European Conference on. IEEE, 2014.

[67] Y. Wu, J. Lim, and M. Yang, "Object tracking benchmark," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1834–1848, 2015.

[68] B. S, Z. Z, W. L, L. D, and Q. L, "Visual object tracking challenges revisited: Vot vs. otb," PLoS ONE, vol. 13, no. 9, 2018.

[69] S. A.W., C. D.M., C. R., C. S., D. A., and S. M., "Visual tracking: an experimental survey," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 36, no. 7, pp. 1442–1468, 2014.

[70] R. O., D. J., K. J. Su H., S. S., M. S., H. Z., K. A., K. A., and B. M., "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, pp. 1–42, 2014.

[71] L. Zheng, Z. Bie, Y. Sun, J. Wang, C. Su, S. Wang, and Q. Tian, "Mars: A video benchmark for large-scale person re-identification," in Computer Vision, 2016. ECCV 2016. 14th European Conference on. Springer, 2016.

[72] E. Real, J. Shlens, S. Mazzocchi, X. Pan, and V. Vanhoucke, "Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video," in arXiv preprint arXiv:1702.00824. arXiv, 2017.

[73] J. Redding, T. McLain, R. Beard, and C. Taylor, "Vision-based target localization from a fixed-wing miniature air vehicle," in American Control Conference, 2006, pp. 2862–2867.

[74] M. J. Monda, C. A. Woolsey, and C. K. Reddy, "Ground target localization and tracking in a riverine environment from a uav with a gimbaled camera," in AIAA Guidance, Navigation and Control Conference and Exhibit, August 2007.

[75] G. Conte, M. Hempel, P. Rudol, D. Lundstrom, S. Duranti, M. Wzorek, and P. Doherty, "High accuracy ground target geo-location using autonomous micro aerial vehicle platform," in AIAA Guidance, Navigation and Control Conference and Exhibit, 2008.

[76] K. Han and G. DeSouza, "Multiple target geolocation using sift and stereo vision on airborne video sequence," in IEEE/RSJ International conference on Intelligent Robots and Systems, no. 5327-5332, 2009.

[77] M. Johnston, "Ground object geo-location usinv uav video camera," in IEEE/AIAA Digital Avionics Systems Conference, 2006, pp. 1–7.

[78] D. Sim, R. Park, R.C.Kim, S. Lee, and I. Kim, "Integrated position estimation using aerial image sequences," IEEE Trans. on Pattern Anal. and Machine Intel., vol. 24, pp. 1–18, 2002.

[79] R. Kumar, S. Samarasekera, S. Hsu, and K.Hanna, "Registration of highly-oblique and zoomed in aerial video to reference imagery," in 15th Int. Conf. on Pattern Recognition, 2000.

[80] D. Hirvonen, B. Matei, R. Wildes, and S. Hsu, "Video to reference image alignment in the presense of sparse features and apperance change," in IEEE Computer Vision and Pattern Recognition, 2001.

[81] D. Linh, R. Satzoda, S.Suchitra, and T. Srikanthan, "Improving robustness of real-time geo-registration process," in IEEE Int. Conf. Neural Networks and Signal Processing, 2008.

[82] H. Li, B. Manjunath, and S. K. Mitra, "A contour-based approach to multisensor image registration," IEEE Trans. on Image Processing, vol. 4, pp. 320–334, 1995.

[83] A. Sappa, A.Restrepo-Specht, and M. Devy, "Range image registration by using an edge-based representation," in the 9th Internation Symposium on Intelligent Robotic System, 2001.

[84] S. Hsu, S. Samarasekera, and R. Kumar, "Automatic registration and visualization of occluded targets using ladar data," in SPIE Laser Radar Technology and Applications VIII, vol. 5086, April 2003.

[85] R. Madhavan, T. Hong, and E. Messina, "Temporal range registration for unmanned ground and aerial vehicles," Journal of Intelligent and Robotic Systems, no. 44, pp. 47–69, 2005.

[86] D. M. S. Jr. and E. N. Johnson, "Indoor navigation for micro air vehicles," in AIAA Infrotech@Aerospace 2010, April 2010.

[87] H. Pottmann, Q.-X. Huang, Y.-L. Yang, and S.-M. Hu, "Geometry and convergence analysis of algorithms for registration of 3d shapes," Int J. Computer Vision, vol. 67, pp. 277–296, 2006.

[88] P. Besl and N. McKay, "A method for registration of 3-d shapes," IEEE Trans. on Pattern Anal. and Machine Intel., vol. 14, no. 2, pp. 239–256, February 1992.

[89] D. Simon, "Fast and accurate shape-based registration," Ph.D. dissertation, 1996.

[90] T. Masuda, K. Sakaue, and N. Yokoya, "Registration and integration of multiple range images for 3-d model construction," in Proc. IEEE Conf. on Comp. Vis. and Pat. Recog., 1996.

[91] J. Canny, "A computational approach to edge detection," IEEE Trans. on Pattern Anal. and Machine Intel., vol. 8, no. 6, pp. 679–698, 1986.

[92] V. Ferrai, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection," IEEE Trans. on Pattern Anal. and Machine Intel., vol. 30, no. 1, pp. 36–51, January 2008.

[93] H. Bay, A. Ess, T. Tuyelaars, and L. Gool, "Speeded-up robust features (SURF)," Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346–359, 2006.

[94] R.Hartley and A. Zisserman, Multiple View Geometry in Computer vision. Cambridge University Press, 2004.

[95] W. H. Press, S. Teukolsky, W. T. Vetterling, and B. Flannery, Numerical Recipes in C. Cambridge University Press, 1992.