Technical Report

Pattern Recognition and Image Processing Group
Institute of Computer Aided Automation
Vienna University of Technology
Favoritenstr. 9/183-2
A-1040 Vienna AUSTRIA
Phone: +43 (1) 58801-18351
Fax: +43 (1) 58801-18392
E-mail: {phil}@prip.tuwien.ac.at
URL: http://www.prip.tuwien.ac.at/

PRIP-TR-128 December 26, 2012

Interactive Tracking of Markers for Facial Palsy Analysis[1]

Philip Limbeck

Abstract

The human face provides a rich source of information which can be exploited to diagnose facial impairments. Facial palsy is one of these impairments, and is caused by restrictions of the nerval actuation of muscles responsible for facial expressions. The main symptoms of this condition are asymmetrical facial movement and partial facial paralysis. To measure its progress, physicians require clinical measures extracted from those locations of the face which provide most information about the facial expression. Small artificial markers indicate these locations by being placed on the patient's face before an evaluation session. A video of the patient is recorded which is used to localize these markers in every frame. This task is currently performed manually by an operator and can take up to 5 hours for a single video. Object tracking refers to the estimation of the position of objects from an image sequence. Illumination and occlusion are considered as the main problems when tracking artificial objects. Natural objects, such as the human face, have a high potential for deformation and are characterized by an irregular texture. As not only one, but multiple markers have to be tracked simultaneously, additional difficulty is imposed by ensuring that markers can be uniquely identified. The thesis explores the possibility of tracking these markers semi-automatically by a applying a sequential Bayes estimation technique, which assesses a set of hypothesis using their congruence with the target model. Hence, the location of each marker can be accurately estimated and occlusions handled efficiently. To improve the accuracy and to reset lost markers, the clinical operator can interact with the tracking system. The results showed that our chosen methods are superior when compared with traditional trackers which use only a single hypothesis concerning the marker locations, while at the same time being able to preserve an accuracy comparable to manual tracking.

[1]Supported by Yll Haxhimusa, Walter G. Kropatsch

Contents

Chapter 1

Introduction and Motivation

Given the evolutionary advantage of human perception, following a moving object seems to be a fairly easy task, even in the context of complex scenes with many other objects and irregularities. On the contrary, even objects of simple shape or appearance can pose difficulties when the human is replaced by a machine. Despite these difficulties, this process is as fundamental to computers as for humans. This process of estimating the position of a set of objects is referred to *Object tracking*. Despite being a relatively young field of research, it has already been employed successfully in a variety of applications [41, pp. 15-26], ranging from surveillance to robotics. However, also the field of medical applications is constantly gaining attention in the research community. Figure 1.1 shows an example of tracking *Escericha coli* bacteria [66].

In this example, bacteria cells are tracked automatically by considering both intensity and motion information. When given a sequence of images, tracking tries to infer the location and movement of an object [41, p. 1]. The selection of the *object* depends on the selected application and may be a car, a person or - like in the example - bacteria cells. Although the problem seems difficult at first, simplifications which are based on constraints concerning motion and appearance allow even barely noticeable, occluded and fast moving objects to be tracked.

1.1 Problem Definition

In medicine, the characteristics of natural objects imply additional difficulties. At first, they are non-rigid, which means they are prone to deform and do not restrict themselves to a small amount of degrees of freedom. Additionally, compared to artificial objects, their appearance is mostly irregular and does not follow a strict texture. Finally, many other anatomical structures are able to occlude the objects of interest at any time, mostly associated with their consistency. As all of these characteristics also apply to the human face, care has to be taken when tracking facial parts. On the other hand, especially because of these issues, the facial structures provide a concentrated source of information, from muscular movement and nerval actuation to characteristics of skin and facial features. This information plays an important role when it comes to diagnosis and treatment of different facial impairments. *Facial palsy* is one of these impairments and concerns a condition where the patient is suffering from asymmetrical facial movement and partial facial paralysis. To quantify this condition, six different facial expressions are necessary to grasp the full capability of facial movement. The facial

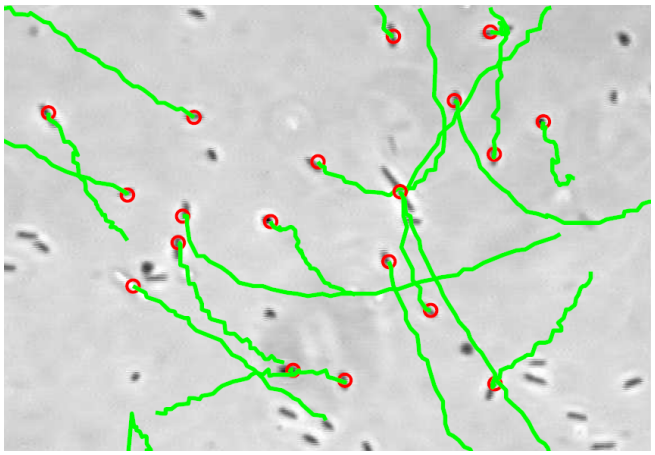


Figure 1.1: Automated tracking of *E. coli* bacteria [66].

expressions are evaluated by means of a set of small artificial facial markers, which are placed on the patient's face. A video sequence is prepared for each facial expression, allowing to locate these markers over time [61]. In each sequence, the trajectories of these markers are used to compare pre- and post-surgical movement. Additionally, if different recordings of the same patient are available, the change of the patient's condition can be analyzed over time. The markers are placed according to a predefined scheme (Figure 1.2), which involves placing three static markers on each *tragus*[1] and on the *rhinon*[2] and 15 dynamic markers around each eye, on either side of the nose and around the mouth. The locations of the markers in the scheme has been determined by clinical reasoning to optimally quantify the condition. This quantification, as currently employed at the *Vienna Medical University* (VMU) has the disadvantage of relying on a manual step to analyze the progress of the disease as well as to compare pre- with post-surgical condition, as the markers have to be located by hand, frame by frame. Depending on the complexity of the patient's movement, this manual procedure can take up to five hours of total analysis time for one patient [61].

An approach which is able to track these landmarks automatically has to deal with both the general problems of object tracking and the application-dependent problems of tracking multiple facial landmarks:

- **Pose**
  Although the head movement is only very slight during the recording session, these changes in pose can still result in appearance changes of the facial markers.

- **Illumination**
  Usually, illumination is a greater problem in outdoor environments, but not a key issue in the presented scenario since the camera is both fixed and indoors. This means that lighting changes can be neglected. However, to gain insight towards the outdoor performance of the chosen methods, an outdoor surveillance sequence is also evaluated.

[1]The tragus is a small cartilage which is located within the ear conch, right before the the auditory canal.

[2]The rhinion is a small cartilage located on the nose. It lies on the border of the cartilage tissue towards towards nasal bone tissue, in the center of the nose.

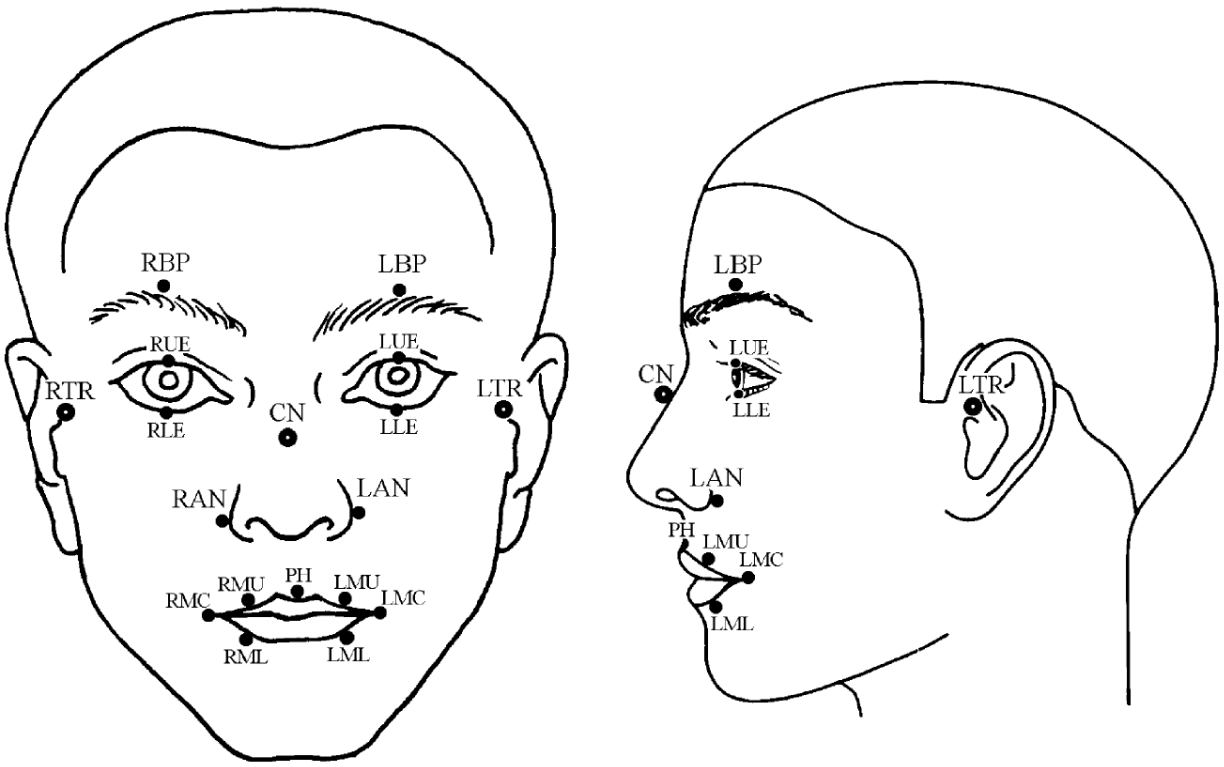


Figure 1.2: Marker scheme which is used at the VMU [20].

- **Noise**
  Noise is the result of perturbation during the signal acquisition progress of the camera. To limit its influence, denoising techniques are employed.

- **Occlusions**
  The most problematic aspect in the presented scenario concerns occlusions. Partial occlusions can be the result of mouth movement, seeming occlusions can be the result of shadowing of the nose, skin color and pigmentation. The face might be partially or totally occluded, for instance due to hair or other facial features, which would make it impossible to locate the affected markers at all.

Additionally, application-specific problems must be considered:

- **Labeling**
  Because multiple targets are tracked in parallel, they can possibly interfere, meaning that labels are prone to change. Additionally, motion or lack of motion of a markers should can lead to a change of the marker's label as well. However, as this labeling is substantial for creating the trajectories necessary for clinical evaluation of the resulting $2D$ locations, counter-measures must be taken to avoid such situations at any cost.

- **Facial deformation**
  Because the face is a non-rigid object, additional degrees of freedom exist, which allow the face to deform. This means that during changes in facial expressions, the appearance of the markers might change as well. Care has to be taken when considering motion as simple translational effect, which would be a deformation of the facial region when considering real $3D$ coordinates.

Although other methods exist to evaluate facial expressions, the object tracking approach has been chosen because there is a wide field of research available and problems are clearly defined and evaluated. Hence the goal and contribution of this thesis is to explore several state-of-the-art methods concerning their applicability on the problem of tracking facial markers. The methods presented in Chapter 5 are compared among each other by using datasets of different marker colors and sizes. These methods differ largely in accuracy, computational resources and their ability to deal with complex situations such as occlusions, changes in appearance and shape. That is why this thesis also discusses these aspects to give a complete and thorough analysis of the methods towards the presented problem (Chapter 6). Implicitly, the previously manual procedure of locating the markers is automated, with the additional possibility to interact with the tracking system when necessary. As it will be shown in subsequent chapters, the analysis time can be reduced by 70.2%, which would mean that given a sequence which would take 5 hours per patient, the estimated time to track the sequence will be 1.5 hours. The accuracy of the tracking schemes, expressed by an Average RMSE of all markers, ranges between 3.8 and 6.9 pixels. Additionally, concerning the best method, only 1.3 % of all evaluated frames needed manual intervention with an RMSE of 3.5583 pixels. The resulting implementation and their evaluation results can be used for future research concerning this topic. To present the results of this thesis to the research community, they have been published at the 21^{st} conference of the International Association for Pattern Recognition (ICPR) [36].

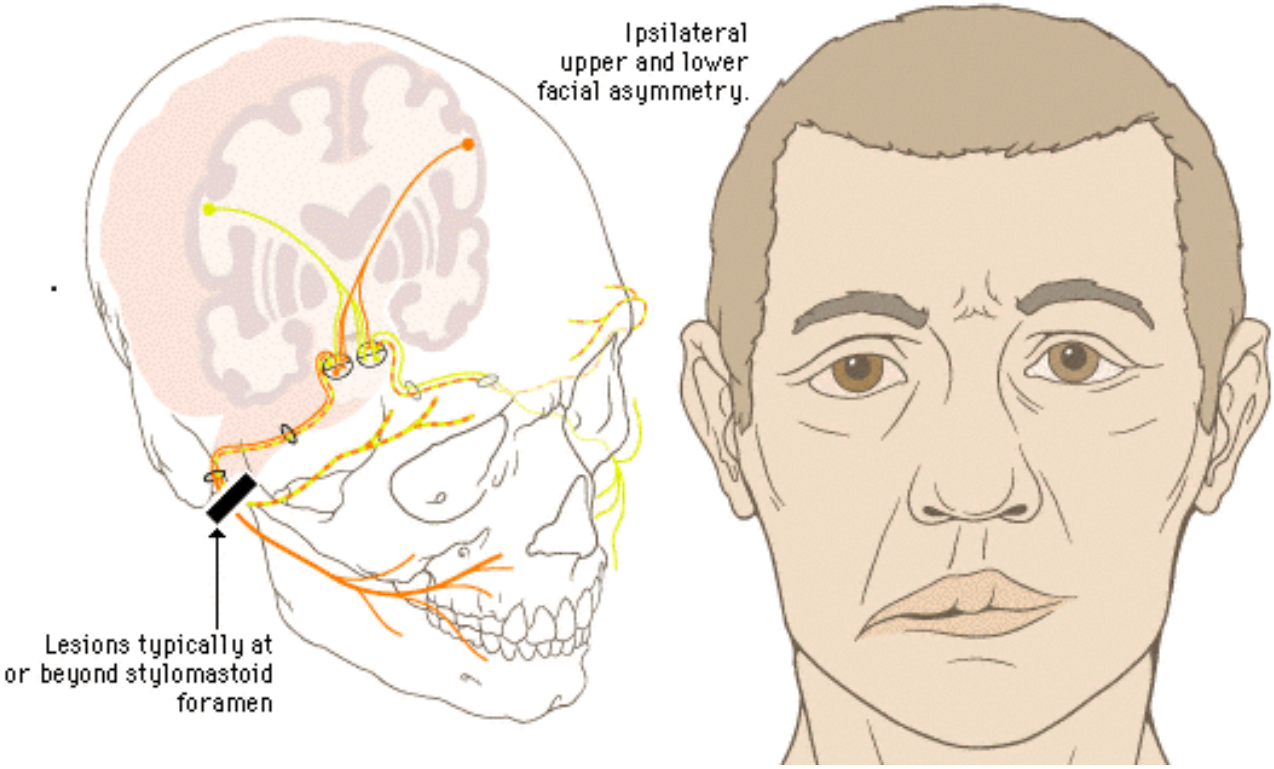


Figure 1.3: Facial nervous system and facial palsy [64].

1.2 Motivation

Facial palsy describes the medical condition of partial or total paralysis of facial muscle tone. Only 25 percent of patients suffering from this condition have known diseases based on infections (e.g. Herpes Simplex), injuries, cancer, autoimmune diseases or fetal malformation. The other cases cannot be tracked back to one of these occurrences and often recover after 3-4 weeks. Symptoms of facial palsy are asymmetrical facial muscle tone often around mouth, forehead and eyelid. The treatment depends on the inducing disease of facial palsy, but often includes application of steroids and antibiotics in strict combination with occupational and physical therapy. Although the aftereffects of facial palsy can remain visible even after treatment (asymmetrical taste-nerve irritation, uncontrolled flow of tears), in many cases a spontaneous recovering can be observed [16]. To diagnose the degree of palsy, grading systems are used to measure the amount of asymmetrical movement when defined facial movements are considered. The most well-known scale is the House-Brackmann scale [26], which subjectively assigns a grade between 1 for normal symmetrical function and 6 for complete loss of facial tone, spasm and contracture. Figure 1.3 shows the primal result of a lesion on either side of the major facial nerves, the *stylomastoid foramen*. The patient suffers from facial asymmetry which in turn causes defects concerning mimic and mouth movement.

The most common cause of facial palsy is Bell's palsy, which is therefore mostly diagnosed for people without associated causes such as tumors, trauma and salivary gland inflammation. The cause

for Bell's palsy is by activation of a virus (*Herpes Simplex*) in the temporal bone behind the ear. In response to this infection, the nerve gets swollen, eventually causing it to degrade. Diagnosis of Bell's palsy is based on a detailed neurologic evaluation as well as evaluation of nose, ear and throat. Occasionally, *Computer Tomography* (CT) or *Magnetic Resonance Imaging* (MRI) scans will help to exclude more dangerous, non-traditional causes of facial palsy. Finally, tests using electromyography or electroneurography (ENoG) determine the final diagnosis of facial palsy. Although the risk factors for Bell's palsy have not been fully understood yet, the incidence is currently 20-25 people per 100.000 per year [16].

Qualitative rating scales such as House-Brackmann, suffer from the disadvantage of having a high intra- and inter-observer variability [14]. In contrast, quantitative methods employ distance measurements to calculate the patient's condition [61]. This quantitative analysis is also employed at the *Vienna Medical University* (VMU), with the disadvantage of relying on a manual step to analyze the progress of the disease as well as to compare pre- with post-surgical conditions. This manual step consists of a placing facial landmarks onto the patient's face using a predefined scheme (Figure 1.2) of clinically relevant locations. The scheme involves placing 3 static markers on each *tragus*[1] and on the *rhinion*[2] and 15 dynamic markers around each eye, on either side of the nose and around the mouth. These marker points are used to grasp the movement of the face over time. To evaluate the clinical condition, different facial expressions are recorded and separated into corresponding sequences. In each sequence, the distances of these markers are used to compare pre- and post-surgical movement. Additionally, if different recordings of the same patient are available, the change of the patient's condition can be analyzed over time. The markers are located manually by hand, frame by frame. However, manual tasks can be both error prone and imprecise, especially after long sessions of the same, repetitive task.

1.3 Preconditions and Requirements

The implemented methods require the markers to be located manually in the first frame. In addition, no entries or exits are modeled, which means that it is assumed that the amount of markers is always constant. Additionally, the following assumptions towards the tracking scenario exist:

- **Limited interaction** between targets - occlusions among markers does not happen.
- **Constant movement**, which means that velocity and acceleration have low variance.
- **No independent movement** of targets, markers only move if other markers move as well.
- **Constant number of targets**, no marker is added or removed, however, it could seem like a marker has been removed caused by occlusion.
- **Small variance of marker appearance**, markers are only allowed to change in small degrees.

[1]The tragus is a small cartilage which is located within the ear conch, right before the the auditory canal.

[2]The rhinion is a small cartilage located on the nose. It lies on the border of the cartilage tissue towards nasal bone tissue, in the center of the nose.

- **Constant scale** Since the scale of the face and of the marker points does not substantially change during the video analysis, handling changes of marker size during a tracking session is not required.
- **Negligible head movement** The initial detection of the face ensures that the search range for later algorithms and methods will be limited to a region of interest which will potentially contain markers or other facial features. Additionally, because the patient is seated and moves his head only slightly (see Section 2.7 for setup), the head movement is neglected and only the absolute movement of the facial landmarks is tracked. However, in other scenarios where the head movement is larger, a compensation strategy has to be integrated to ensure that only the relative movement of the markers is tracked.

1.4 Organization of this Thesis

Chapter 2, discusses state-of-the-art concerning both existing methods for diagnosing the degree and progress of facial palsy and provides an introduction to other approaches dealing with tracking facial features beyond the medical context. Chapter 3, introduces the state-of-the-art for object tracking in general. The available features are presented as well as possible approaches to represent the target. Also, localization methods are discussed which allow creating correspondences from one frame to the next using the described features and appearance representations. Since Bayes tracking is the central approach in this thesis, a separate Chapter 4 deals with the theoretical background of Bayes tracking and their different peculiarities. In Chapter 5, the chosen methodology is presented. This chapter also discusses the expected behavior of both the selected baseline methods and comparative Bayes methods. An evaluation of the presented approach is provided in Chapter 6. Since it is necessary to magnify different aspects of the implemented approaches, the evaluation is split into different parts. These include single target object evaluation and multi target tracking of facial markers. Finally, a conclusion is given in Chapter 7, which reflects the discussed results and tries to identify advantages and disadvantages of the evaluated tracking schemes, both in terms of accuracy and computational resources.

Chapter 2

Analysis of Facial Characteristics

In this chapter, a chronological evolution of methods which deal with trying to grade the progress of facial palsy or facial asymmetry are presented. In general, we can distinguish between *subjective* grading methods, which try to infer progress from asymmetry measures and *objective* methods, which try to measure the differences among prominent facial location. The major problem associated with subjective methods is their high intra- and inter-observer variability [14]. On the contrary, *objective* methods (Section 2.2) try to overcome the problem of high intra- and inter-observer variability by measuring the differences among prominent facial locations. Early methods only considered two images (Section 2.3). Additionally, Computer Vision topics have not been popular and mature enough to compete with manually located landmarks or facial characteristics. Especially the lacking accuracy made several hours of manual pinpointing the preferred method. Recently, semi-automated methods have been explored which try to reduce the time necessary to locate facial landmarks or features by hand at different rates. To generate understanding for other possible approaches, not only methods towards medical context, but general method for facial feature tracking (Section 2.4) are discussed. Finally, the current workflow at the Vienna Medical University is outlined (Section 2.7).

2.1 Surface Electrodes

In the early history of facial palsy analysis, surface electrodes had been attached on the face and the regional differences between electric current have been used to measure asymmetry and muscular actuation [49].

2.2 Supervised Methods

Instead of tracking single points, supervised methods can be used to classify movement patterns according to a defined quantitative grading scale such as House-Brackmann [26]. The patients are asked to perform defined facial expressions which are recorded. In every frame, the first task is to remove any in-plane rotations by such that the facial mid-line is perpendicular in the image [24]. Block-matching is used to compute the affine transformation necessary remove rigid head motions such that only non-rigid facial expressions are left. After detection of each frame with the facial expression's apex, which corresponds to the maximal facial response, the image is cropped into regions of interest. The regions

of interest are defined according to the facial expression, which is for instance the mouth for a smiling expression. Multi-resolution Local Binary Patterns (Section 3.1.2) are extracted along the regional frames around each apex frame and used to compute block-wise histograms describing the motion. Instead of only computing Local Binary Texture descriptors along the spatial domain, additionally the temporal domain is considered. These histograms are then used to compute symmetry measures using the Resistor-Average (RA) distance, which is based on the Kullback-Leibler (KL) divergence:

$$D_{RA}(p,q) = [D_{KL}(p \parallel q)^{-}1 + D_{KL}(q \parallel p)^{-1}]^{-1} \tag{2.1}$$

with

$$D_{KL} = \sum_{x} p(x) \log(\frac{p(x)}{q(x)}), \tag{2.2}$$

representing the difference between the two probability distributions $p(x)$ and $q(x)$. In this case, the two distributions are the combined set of histograms extracted from the local binary patterns in the different regions of interests on either side of the face. The Resistor-Average distance is used to address the problem of the missing symmetry property of the Kullback-Leibler divergence. These symmetry measures based on the Resistor-Average distance are finally fed into a classifier such as a SVM (Support Vector Machine) to map them to a House-Brackmann grade.

2.3 Dual Image Analysis

The methods presented in this section rely on two images to evaluate the condition of the patient. One image shows the patient's face at rest, while the other one shows the apex frame, that is, the maximum response of the muscles responsible for the corresponding facial expression.

2.3.1 Maximal Static Response Assay

In the Maximal Static Response Assay (MSRA) [28], a video of the patient is recorded. From this video, two frames are selected which correspond to the face at rest and its maximal response. After digitization, these frames are then used to manually locate the centers of the facial markers. From these information, distances can be calculated manually, corresponding to the facial asymmetry.

2.3.2 Moiré Method

An approach using Moiré patterns has been presented by Yuen et al [32]. The Moiré method is used to project a black-white striped pattern (Figure 2.1) onto the patients face. Three indices indicate the progress of facial palsy:

- *Inner canthus moiré index (IMI)*: Difference between right and left number of bending moiré stripes, performed between the inner canthus and the forehead
- *Oral angle moiré index (OMI)*: Difference between right and left number of moiré stripes from the sub-nasal point to the mouth corner

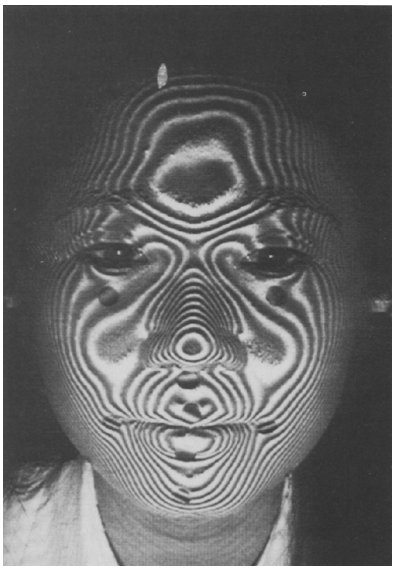


Figure 2.1: The projected Moire pattern on a patient [32].

- *Nasolabial groove moiré index (NMI)*: Number of bending moiré patterns in the nasolabial groove

In their paper, they point out that these three indices correlate with the House-Brackmann scale concerning the degree of facial palsy progress, and can be used instead of subjective grading. This method uses a single still image to automate a subjective grading scheme based on the Moiré stripes. It does not use any temporal information such as magnitude or direction of change. Healthy subjects result in an NMI above 90 percent and OMI and IMI close to zero, while pathological cases have an OMI up to 18 percent [32].

2.4 Tracking

The methods in this section use tracking methods which are partly described in Chapter 3. Instead of merely analyzing two pre-selected images, the complete video of a patient is evaluated.

2.4.1 Optical Flow

Park et al. [46] presented an approach of analyzing facial palsy in an uncontrolled environments such as casual home web-cams with resolutions down to 640×480. The patient does not have to be in a controlled environment, but can be at home using its own web-cam. Bi-literal lip points are tracked using Lucas Kanade (Section 3.1.3) point tracking, eventually creating a trajectory plot of these points. White balance correction and transformation into HSV space is used to limit the influence of environmental lightning conditions. Lips are then segmented by finding color space centroids using fuzzy C-means and region growing in HSV space. Since no Gaussian model is used to consider changes in lighting conditions, the robustness against such factors are very poor, which would have been important in the considered environment. The method achieves a mean and standard deviation of

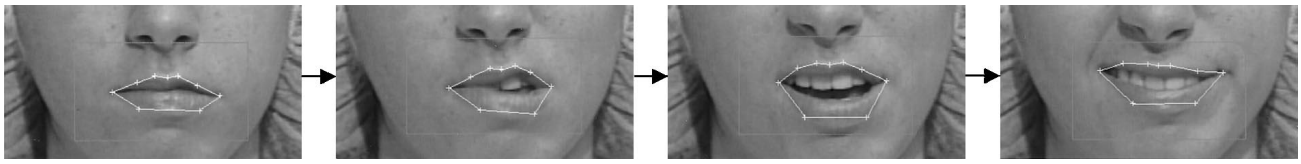


Figure 2.2: Tracking facial features with sparse optical flow [62].

5.4 ± 8.89 pixels over all evaluated frames for a 7×7 search window on 320×240 images and 1.86 ± 2.63 pixels for a 14×14 search window, representing the minimum error for the given setup between automatic tracking and the ground truth.

Automated Facial Analysis (AFA) [35] is a combination of different optical flow based methods to detect and classify facial expressions into so called Action Units (AU). The facial feature tracking part, which uses sparse optical flow tracking [39], has been used to track facial feature points without facial markers (Figure 2.2). The study in [62] which compares the manually operated MSRA [28] with AFA, determined a high correlation between these methods, with the advantage of no manual intervention during tracking. To allow a certain pose change of the head, affine or perspective transformations are introduced to reverse changes resulting from different head pose. Pyramidal sparse optical flow tracking [7] is used to track the points according to different scales, using simple Gaussian pyramids and coarse-to-fine tracking. In their experiments 18 points have been examined. They can be subdivided into 9 blue markers, 7 anatomical landmarks and 2 calibration dots used for scaling. 16 points are manually marked in the first frame and tracked using AFA over the subsequently arriving 75 frames in average. Since there are no explanations on data associations, the markers are unlabeled and prone to interfere when markers join and disconnect, especially in the mouth region. On the other hand, occlusions are handled implicitly by the chosen sparse tracking method. Another problem of not using fixed landmarks is the error induced by tracking not always the same facial features among different sessions of the same patient.

2.4.2 Active Appearance Models

A different approach uses *Active Appearance Models* (AAM) to model the appearance changes of facial features during movement of the mouth region. Instead of tracking the positions directly and using them for clinical diagnosis, the difference of movement between the left and right corner of the mouth are computed. Based on this difference a classification is given using a fixed scale. The scale is then mapped to the grades I to VI of the House-Brackmann grading scale (Table 2.1). The AAM are used to track the corner points of the mouth. To evaluate the proposed system, five healthy subjects with normal and smiling expressions are used and their smiles were synthesized by creating different levels of asymmetrical interpolation corresponding to the levels I to VI of the House-Brackmann grading scale. These images are then put into the corresponding House-Brackmann grading classes manually for comparison [15]. The major disadvantage of the approach is that the clinical assessment is limited to the evaluation of the mouth region, although other facial regions are also necessary to give a full statement of the progress [61]. Additionally, evaluation is based only on synthetically modified images to create pathological cases out of normal subjects, instead of using pathological data.

Grade	*Description*
I	Normal symmetrical function in all areas
II	Slight weakness noticeable only on close inspection
III	Obvious weakness, but not disfiguring
IV	Obvious disfiguring weakness
V	Motion barely perceptible
VI	No movement, loss of tone, no contracture or spasm

Table 2.1: House-Brackmann grading scale [26].

2.4.3 Gabor Jets

In an approach described by Zhu et al. [69], 28 facial features are automatically detected and tracked. These features and their corresponding vicinity are represented by a vector of the convolution results of 60 Gabor kernels. The feature vector describing each facial feature is used both for initial feature detection and for tracking. Initially, face, eyes and mouth are detected using an AdaBoost face detector and a trained facial mesh which is extracted from the mean face is used to compute a first approximation of the facial features. The mean face has been extracted from a set of frontal faces of different people, however it is neither mentioned how many nor under which conditions. In a second step of detection, the Gabor jets of the approximated positions are extracted and searched in a training set of Gabor vectors of different expressions, conditions and individuals. This step is repeated until convergence and completes the detection step. For tracking, Kalman filtering (Section 4.3) is employed, which prediction step consists of searching for the vector in a local search region using a Gabor phase estimation technique. The displacement vector $\boldsymbol{d}$ is estimated by performing a second-order Taylor expansion of $\cos(\phi_j - \acute{\phi}_j - \boldsymbol{dk}_j)$ around the the parameters $J(\boldsymbol{x}) = \left\{m_j e^{i\phi_j}\right\}_{j=1:n}$ and $J(\boldsymbol{y}) = \left\{\acute{m}_j e^{i\acute{\phi}_j}\right\}_{j=1:n}$:

$$S(J(\boldsymbol{x}), J(\boldsymbol{y})) \approx \frac{\sum_j m_j \acute{m}_j [1 - 0.5(\phi_j - \acute{\phi}_j - \boldsymbol{dk}_j)^2]}{\sqrt{\sum_j m_j^2 \sum_j \acute{m}_j^2}}, \tag{2.3}$$

where $\boldsymbol{k}_j$ is the wave vector of each Gabor kernel, $\acute{m}_j$ and m_j is the corresponding scale parameter, and $\acute{\phi}_j$ and ϕ_j is the corresponding orientation parameter [69]. The facial feature vector is updated using the Gabor vectors of the new position to deal with illumination changes over time. Because this induces drift, the positions are corrected using neighborhood search in a different training set incorporating not only frontal faces but also different face orientations. To correct the facial pose, it is first estimated by adapting a generic 3D model to the individual 3D model using 7 rigid points that do not change much under facial expressions. The pose-eliminated 2D points of the facial features are then constrained using a PCA-learned shape model. Although a complete system is presented, the facial shape mesh is unsuitable for clinical conditions because no pathological cases are modeled. Additionally, the effort is very high to collect data in advance and create the necessary models. On the contrary, a complete system is presented, which is highly robust by reaching an average error of 1.89 ± 0.89 pixels.

2.4.4 Mean Shift

An approach using Mean Shift 3.3.1 has been presented by Barker et al. [4]. Seven markers with four being around the mouth, one on the chin, one on the rhinion and one between the eyes are tracked using Mean Shift. The first frame is used to provide manually $l \times l$ sized appearance templates. The corresponding region of interest (ROI) of the tracker is then defined as $l + \Delta \times l + \Delta$, where Δ describes additional size. The luminance of each pixel in the ROI is calculated and a threshold is applied to convert the ROI into a black and white mask for

Figure 2.3: Optical Strain Map of smile expression [55].

which the center of mass is calculated using Mean Shift. Because this threshold is found empirically to match the appearance of the marker it is prone to fail. Because these failures are expected to happen by the authors, a confidence measure based on a pixel ratio towards the marker appearance is used to determine tracker break down. After such failures have been detected a Gaussian collocation model (Equation 2.4) is used to estimate the position of missing marker x_{tm} at time t from the known marker positions x_{tp} at time t:

$$x_{tm} = -P_{mm}^{-1} P_{mp}(x_{tp} - \mu_p) + \mu_m, \tag{2.4}$$

where the inverse covariance matrix P is reordered according to the coprecision of missing locations with missing locations P_{mm} and missing locations with present locations P_{mp}. The mean vector is divided into missing μ_m and present μ_p as well. Although this approach seems to fail very often, it still performs robustly using the presented collocation model. However, as results indicate, the performance is highly dependent on the chosen training window of the statistical collocation model. It performs best when an online model is used which considers the last 20 frames of all locations being known.

2.4.5 Optical Strain Maps

Instead of tracking single points, dense optical flow methods in combination with strain maps [55] can be used to describe the movement of the underlying facial muscles. Based on existing dense optical flow information represented by a vector $[p, q]^T$, optical strain vectors $[u, v]^T$ can be computed by defining the interval between subsequent frames Δt from the existing optical flow vectors:

$$\frac{\partial u}{\partial x} = \frac{\partial p}{\partial x}\Delta t, \frac{\partial u}{\partial y} = \frac{\partial p}{\partial y}\Delta t, \frac{\partial v}{\partial x} = \frac{\partial q}{\partial x}\Delta t, \frac{\partial v}{\partial y} = \frac{\partial q}{\partial y}\Delta t \tag{2.5}$$

These strain values are finally summed up and fed into an expression localizing algorithm which subdivides the video into several apex frames by analyzing the provided strain magnitude. These frames are then used to calculate the final strain maps, representing the amount of deformation due to muscular contraction (Figure 2.3). The available strain maps can be compared with earlier data, allowing to grasp changes in muscular movement after surgery. Although a dense information is given about the muscular movement, the disadvantage of the provided method is the high dependency on robust and accurate optical flow data, which is difficult to compute in the highly homogeneous regions of the face.

2.4.6 Facial Contour Extraction

Instead of tracking or extracting points, contours can also be used to quantitatively evaluate the state of facial palsy. Facial feature contours can be extracted by applying dynamic thresholding and center of gravity extraction

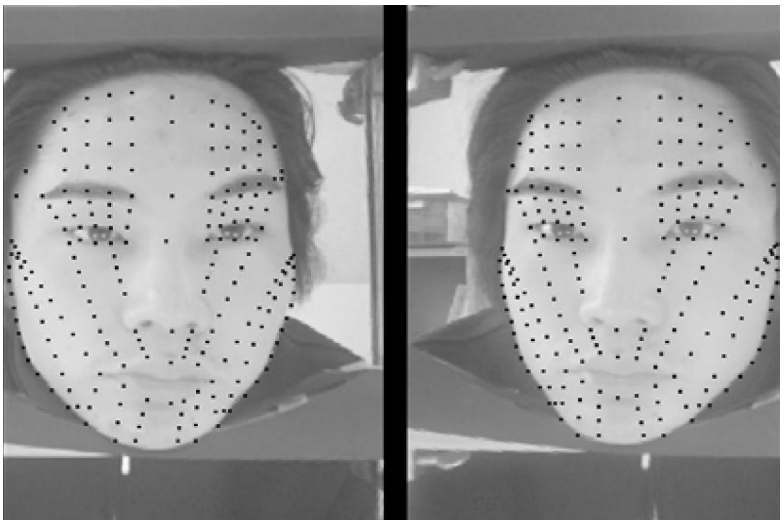


Figure 2.4: Stereo images showing facial feature points [52].

to locate the eyes. Color spaces like *YIQ* (Luminance, In-Phase and Quadrature) or *HSV* can be used to robustly extract these features without being influenced by changing perceptive chromaticity. Godlove Differencing [58] is used to extract the contours of eyes after locating them, while lips are extracted using a linear transformation resulting in control points which will be used in spline interpolation to have the final lip contour (Figure 2.4). Finally, different ratios are calculated which define the index of facial palsy progress. Although the method is able to differentiate between pathological and non-pathological cases, not all medically relevant facial regions are considered e.g. nose. No evaluation is given concerning the correlation with the subjective grading scales. This method has been extended to 3D [52] by using a stereo vision camera. The images are distorted using a circular calibration image.

Based on the sum of the differences between the rest and apex frame on either side of the face are computed and used to determine a facial palsy score based on single scores from cheek, mouth and eye [52]. In this 3D approach, only healthy persons have been studied and the system was not adapted to their 3D extension. Additionally, scars and other disturbing facial objects might prevent the successful detection of the contour.

2.5 Marker-based Localization

Apart from classical systems where markers of different colors are used e.g. white [27], other markers are coated with retro-reflective material to reflect the impacting light back to its source. With this approach, localization of markers becomes easier due to greater contrast with the rest of the facial background. Additionally, it cannot be confused with existing facial structures such as moles or acne. In these cases, simple thresholding methods can be used to segment the markers from the rest of the face. The diameter of facial markers can vary from between 2 to 6 mm. Using a different space in the light range, such as infra-red, the possibly noisy or cluttered background around the marker locations can be ignored completely. Additional types of markers even emit a detectable signal for them [47]. Still, in the literature, no automated handling for occlusions is presented for these systems. In case of semi-automatic correction of missing markers, another footage using visible light has to be utilized to allow manually tagging missing markers in case of error or occlusions.

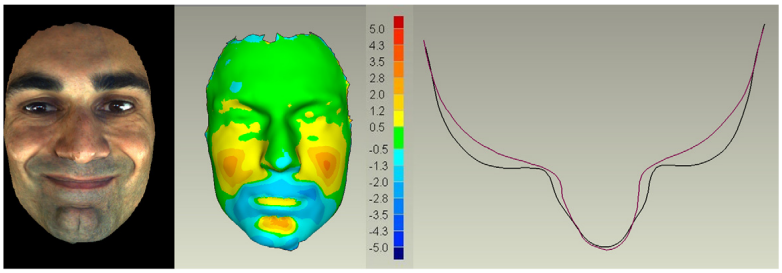


Figure 2.5: Registered facial surface and its asymmetry plot [47].

2.6 Marker-free Localization

In contrast to landmark based systems, the facial surface is completely registered and allows more detailed analysis of muscular movement over time. Because neither single facial features nor landmarks are tracked, the reproducibility of feature selection or standard deviation concerning landmark placements are not an issue. In addition to trajectory evaluation methods, asymmetry information can be computed by comparing the surface profiles and their curvature properties (Figure 2.5). The major disadvantage is the high computational cost concerning registration and subsequent rendering of the data [47].

2.7 Marker-based Diagnosis with Manual Localization

Another clinically relevant approach has been introduced by Frey et al. [61] and is currently used for evaluation facial palsy progress at the Vienna Medical University (VMU). This section describes each step of this workflow in chronological order.

2.7.1 Patient Adjustment

The setup for evaluation consists of a device which uses two mirrors on either side of the patient (Figure 2.6), to generate *virtual cameras* which have the same intrinsic properties than the camera used for recording. This allows generating different views of the patients face without having to compute the relations between three proper cameras. Before calibration, the patient's chair is adjusted properly to accommodate his height and facial location. Afterwards the patient is removed from the scene and the calibration device is used to calibrate the camera.

2.7.2 Calibration

The Software *Pinnacle Studio*® is used for calibrating the camera by requiring the selection of calibration points. The yellow calibration points on the calibration device in between the mirrors (Figure 2.6) have a diameter of 1.8 millimeters. Subsequently, the intrinsic and extrinsic camera parameters are computed using the correspondences between the two virtual cameras and the proper camera.

2.7.3 Marker Placement

Two types of markers are distinguished. Three reference markers are used to identify the other type of markers across several sessions over time on the same patient. The position is determined using a clinical nose measurement device. The reference markers consist of yellow dots of 3 to 5 millimeters diameter. The second type of markers is used for medical diagnosis and of different color. They can be of any color, although black or brown

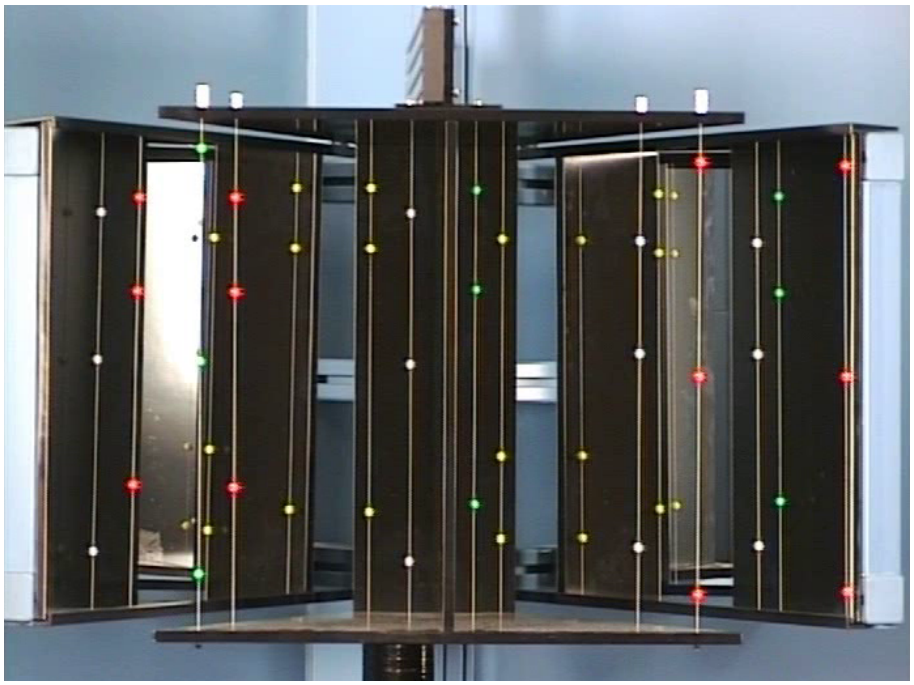


Figure 2.6: Calibration device with calibration points [61]. (Courtesy of VMU)

markers seem to be preferred. The markers are drawn based on a medical scheme which reflects the condition of the patient. Each position has its own named identifier. The size of each marker is given by the pen used for drawing. In case of possible occlusion and the mouth corner, bigger markers or even lines are drawn to ensure correct crossing points with important facial features such as corner of the mouth or eyebrows. Markers on the mouth corner are drawn while the mouth is open. The markers below and above the eye are determined by using the line upright through the patient's pupil. Due to facial movement, the preferred method is that markers should be drawn instead of pasted, however, the smaller the pasted marker gets, the less this will become and issue.

2.7.4 Record Video

The evaluation video is recorded at a resolution of 720×576 pixels. Subsequently, the patient is asked to do a set of ten mimic actions, for example raising the eyebrows or smiling while showing the teeth (Figure 2.7). Each of these actions is repeated three times to allow selection of the best sequence for each mimic action. It is important for the patient to behave not too fast or too slow, on the one side to avoid motion blur and on the other side to prevent additional time to analyze the frames. The patients mimic action usually starts with a point at rest with total muscle relaxation. The action then rises to a plateau of maximal muscle tension and then falls back to relaxation. In addition to video information, a microphone is used to record audio information as well. The patient is asked to say both his name and address, although this vocal audio information is neither technically nor medically evaluated.

2.7.5 Static Photos

In addition to the movie, static photos are made for each mimic action of both rest and maximal tension. This is basically to support the analysis process, but no real requirement.

Figure 2.7: Patient performing 'smile-while-showing-teeth' expression. (Courtesy of VMU)

2.7.6 Localization

The resulting video footage is now tagged manually using custom software. Using the mouse, an operator has to click on every marker in every frame. however The manual positioning causes a variance of the locations from 2 millimeters up to 1 centimeter over one video. During tagging, manual point selection is based on four times enlarged sub-frames which allow positioning the centroid point of the marker or calibration points. Although the software offers automatic tagging of the next 5 frames, it is rarely used because of its high error rate. For one mimic action, 1 to 2 hours of work is required in total for tagging each frame. To reduce the amount of frames during low facial movement i.e. in case of plateau frames, only every 3rd frame needs to be tagged.

2.7.7 3D Reconstruction

3D Reconstruction is based on the manually localized calibration points of the different views. As these mirror views actually form a set of equal cameras in terms of intrinsic properties, only the extrinsic parameters, i.e, the parameters necessary to map the their relative 3D transformations to each other, need to be determined [42]. The calibration points which are manually selected in the mirror views of the calibration device allows inferring the necessary transformation matrix to relate these points and reconstruct them in 3D [61].

2.7.8 Evaluation

The evaluation workflow usually starts with selecting the date of the video session, then the facial expression and finally the type of visualization, which could be a 2D, a 3D plot or a specific medical chart. Concerning medical relevance, the closure of eye and smile is most important to determine the facial palsy progress. The provided visualizations and metrics could be - among others - comparison plots of left and right side or trajectories of single points. The L^2 norm is used instead of the the geodesic distance, because after surgery the muscle surface and volume would change in a way that the geodesic distances are no more comparable. The most important metric is the static asymmetry which is computed temporally between the facial rest and its maximum tone on both sides. Spatially, the static asymmetry is computed from the left and right distances between each tragus -

which is the small cartilage which is located within the ear conch, right before the the auditory canal - and the corresponding mouth point. For two selected markers, comparison graphs can be created which compare time vs. expressive amplitude from either side of the face.

Chapter 3

Building Blocks for Object Tracking

In Computer Vision, *tracking* is the process of following one or many objects by incorporating information from the past. In contrast to mere object detection, which is about recognizing the object in the image. Tracking deals with the selection of a model to represent the object and the search for its best match within the image. Tracking methods utilize image features and motion information to estimate the position of a target in the next frame x_{t+1} based on the location in the current frame x_t. Object tracking consists of two not necessarily distinct components. At first, the targets must be located (manually or automatically) and represented accordingly, which is highly dependent on the specific field of application. This is often modeled as a bottom-up process, meaning that no predicate knowledge about the image is necessary. Only low-level features, such as image intensity or gradients are considered and used to form high-level object models. As a second step, filtering and data association processes are required to deal with the dynamical aspects of the tracking process by evaluating different possible hypotheses concerning the object's properties. In contrast to the first step, this is mostly modeled as top-down approach, having additional prior information, such as motion, available [13]. A taxonomy [67] of the state-of-the-art tracking methods is depicted in Figure 3.1. In general, tracking can be subdivided into kernel tracking, silhouette tracking and point tracking methods. This thesis focuses on kernel tracking methods such as Monte Carlo tracking (Section 3.3.2) and Mean Shift (Section 3.3.1). As baseline approach, a point tracking method has been selected which is described in detail in Section 3.3.1. This chapter gives an outline on the evaluated tracking methods and their main theoretical methods they are based upon. Starting with the features (Section 3.1) which can be considered for tracking, the chapter continues with briefly describing the most important methods of appearance representation (Section 3.2). Subsequently, different methods of localization are described and how they deal with the associated tracking problems (Section 3.3). Their advantages and shortcomings are discussed as well, comprising a thorough overview of methods used during the experiments and related methods of importance.

3.1 Features

The performance of the tracking scheme largely depends on the quality of the selected low-level features. This section is intended to outline the features used for representing the model of the tracked objects (Subsection 3.1.1 and Subsection 3.1.2). Additionally, this section explains the theory behind estimating motion between two successive frames (Subsection 3.1.3), to establish the background for the deterministic component used in the evaluated tracking schemes.

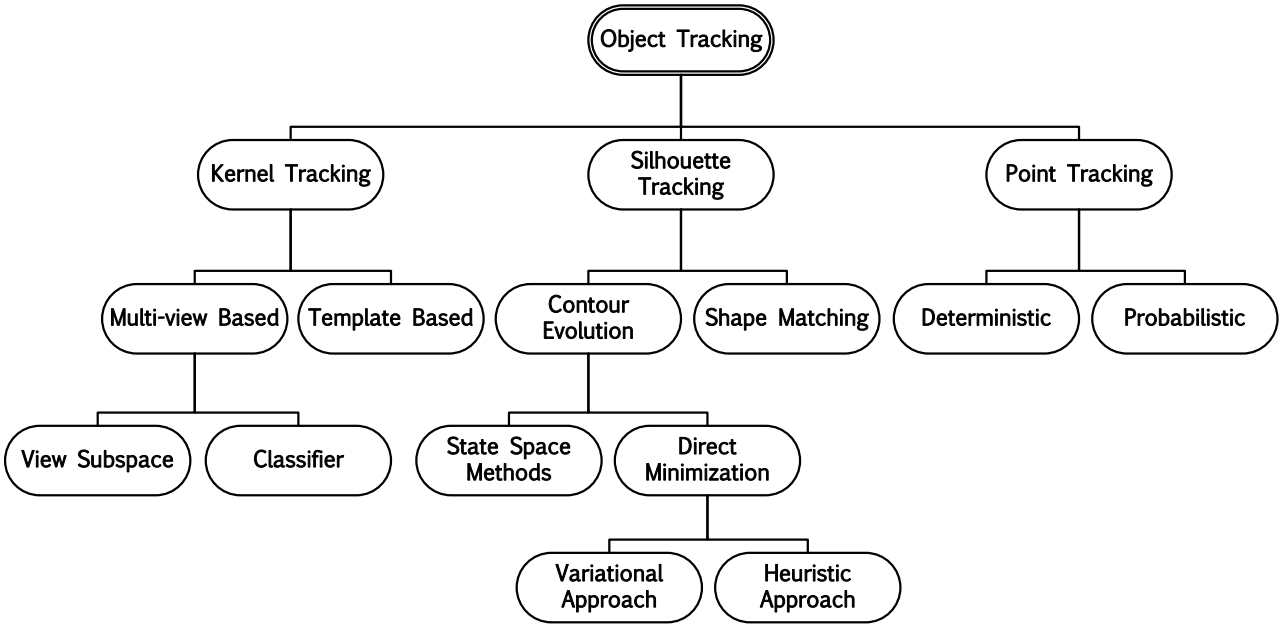


Figure 3.1: A taxonomy of object tracking methods[67].

3.1.1 Color

Different representations of color exist. Their difference is mainly experienced in the possibility of human perception and the effect of changes in the environment without changing the emitted color of the object [41, pp. 32-33]. This subsection is intended to give on overview about the used color spaces.

RGB

The *RGB* color space uses a 3D Cartesian coordinate system to encode color. Its base vectors represent the colors *Red*, *Green* and *Blue*. All other colors are mixture values in the given coordinate system, where $[0, 0, 0]$ represents black and $[1, 1, 1]$ represents white. The main diagonal with equal values on all 3 axis represent different shades of grey. [41, pp 35-36]

HSV

The *HSV* color space is a non-linear transform from RGB. Instead of a Cartesian coordinate system,a cylindrical coordinate system is employed. The angle around the base of the cylinder represents the Hue value, the distance from the center represents the Saturation and the distance from the bottom of the cylinder represents the Value. In this case, black is represented by anything with a Value of 0. White is represented by anything with a Value of 1. *HSV* is used when lighting changes can change the perceived color, although the emitted color information stays the same. By suppressing the Value channel, the emitted color can be preserved [41, pp. 37-39].

YCbCr

The *YCbCr* color system is linear and provides separation of luminance and chrominance information. This makes it suitable to segment a facial region from its non-facial background. Y represents the luminance component and C_b and C_r represent the blue-difference and red-difference components of chroma, respectively. Instead

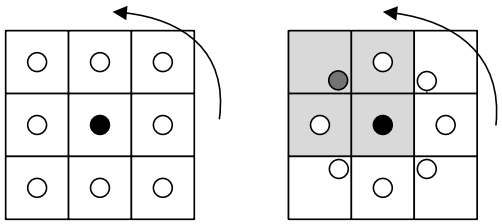


Figure 3.2: Comparison of original $LBP_{8;1}$ with extended $LBP_{8;1}$.

of representing an absolute color value, its value information is relative to the original RGB signal. The color system can be modeled as an affine transformed box with one corner of the box being black (lowest value of Y) and the opposite corner being white (highest value of Y) [41, pp. 36-37].

3.1.2 Local Binary Patterns

Descriptors such as *Local Binary Patterns* (LBP) [44] are used to describe the local texture of an image. LBP encode the textural properties of each neighborhood in the image. The operator $LBP^{riu2}_{P,R}$ is able to encode Grey level information at different values of P, which describes the quantization of the angular space, and R which defines the radius around the center pixel. It is invariant to Grey-scale changes and, using an extension of the operator, invariant to rotations. The operator is based on the joint distribution of Grey values around a center pixel g_c:

$$T = t(g_c, g_0, ..., g_{P-1}). \tag{3.1}$$

For invariance to changes in Grey-scale, the center pixel is subtracted without losing any information:

$$T = t(g_c, g_0 - g_c, ..., g_{P-1} - g_c). \tag{3.2}$$

It is assumed that the resulting distribution of differences are independent of g_c itself, which results in the following factorization of the distribution:

$$T \approx t(g_c)t(g_0 - g_c, ..., g_{P-1} - g_c). \tag{3.3}$$

This results in a small loss of information. However, as it is necessary to achieve the invariance because $t(g_c)$ describes the overall luminance of the image, this does not provide useful information about the texture, so it will be set to 1 instead. An additional step is to consider only the signs of the differences instead of their values:

$$T \approx t(\Theta(g_0 - g_c), ..., \Theta(g_{P-1} - g_c)), \tag{3.4}$$

where Θ is the Heaviside function. The final LBP value at the given position is then computed by assigning a binomial factor 2^p to each sign difference $\Theta(g_p - g_c)$. Hence, the resulting operator value is then computed by:

$$LBP_{P,R} = \sum_{p=0}^{P-1} \Theta(g_p - g_c)2^p. \tag{3.5}$$

Originally, only the 8-neighborhood around each center pixel has been considered. However, due to its limited spatial support it has been extended very early to support different radii and neighborhood sizes. Figure 3.2 compares the original LBP approach with the extended approach which uses interpolation. While in the original LBP approach (Figure 3.2 left), only the values of the 8-neighborhood (white circles) around the center pixel (black circle) are considered, the extended approach (Figure 3.2 right) allows positioning values of interest (white

circles) at different radii around the center pixel (black circle). Each Gray value (gray circle) is retrieved by interpolating its four neighboring pixels (gray background). The values are indexed counter-clockwise, starting with the right-most value. The angular coordinate system around the center pixel at $[0, 0]$ is defined by $-R\sin(2\pi p/P)$ and $-R\cos(2\pi p/P)$. Interpolation methods such as bicubic interpolation are necessary to compute the Grey values at the corresponding circular points around the center. Because rotation of signs of the circular neighborhood points will result in different LBP result values, rotation invariance can only be achieved if the most-significant bits of the encoded LBP value do not change. Rotation-invariance can be achieved if the following extension is applied:

$$LBP^{ri}_{P;R} = min\left\{ROR(LBP_{P;R}, i) \mid i = 0, 1, ..., P-1\right\}, \tag{3.6}$$

where $ROR(x, i)$ shifts the P-bit number x circular bit-wise to the right i times. In this case, the resulting number will always have the same value, regardless of the rotation. If Grey-scale invariance is not required and also the contrast should be incorporated, the rotation invariance measure of local variance:

$$VARLBP_{P,R} = \frac{1}{P}\sum_{p=0}^{P-1}(g_p - \mu)^2, \tag{3.7}$$

where μ is defined as $1/P\sum_{p=0}^{P-1} g_p$, can be used to encode the variance of the pixels in the neighborhood to describe the statistical properties of the contrast. By combining the output of the LBP operators with different values of P and R, a multi-resolution approach can be implemented elegantly. The problem of small spatial support area is solved by multi resolution LBPs, which are also able to incorporate of Gaussian pyramids. Usually, the output is encoded as LBP occurrence histogram, however, to encode the spatial information of the LBP as well, an LBP image can be created [45]. LBP-based textual information has been used successfully in different tracking approaches, especially for face tracking [51].

3.1.3 Optical Flow

Optical flow is the measure for temporal or spatial motion. While initial methods only considered difference methods or correlation, recent methods are based on optimization methods of the image's gradient vector field. Computing optical flow is not limited to Grey values, but can also be computed from other features, such as SIFT [37] or Wavelets [65].

In its simplest form, a template image $I(\boldsymbol{x})$, is shifted over a second baseline image $I_0(\boldsymbol{x})$, where the template image is usually smaller than the baseline image. To find its displacement vector $\boldsymbol{u} = [u, v]$, the solution to this problem can be expressed as least-squares equation, the *sum of squared differences* (SSD):

$$E_{SSD}(\boldsymbol{u}) = \sum_i [I(\boldsymbol{x}_i + \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2, \tag{3.8}$$

with i being the corresponding index within the images I and I_0. In case of temporal instead of spatial movement, the problem will increase in complexity, since there is not one single displacement vector, but the displacement vector $\boldsymbol{u} = (u, v)$ will become the location-dependent displacement vector $\boldsymbol{u(x)} = (u(\boldsymbol{x}), v(\boldsymbol{x}))$ which results in a location dependent energy function:

$$E_{OF}(\boldsymbol{u}_i) = \sum_i [I_t(\boldsymbol{x}_i + \boldsymbol{u}(\boldsymbol{x})_i) - I_{t+1}(\boldsymbol{x}_i)]^2. \tag{3.9}$$

In this case, the error is not minimized spatially between one template image and one candidate image, but two images with the same size but different temporal locations.

In general, motion induces intensity changes. Exceptions are given for example in case of a rotating sphere with a homogeneous surface. In this case, the perceived optical flow is zero although the sphere is obviously

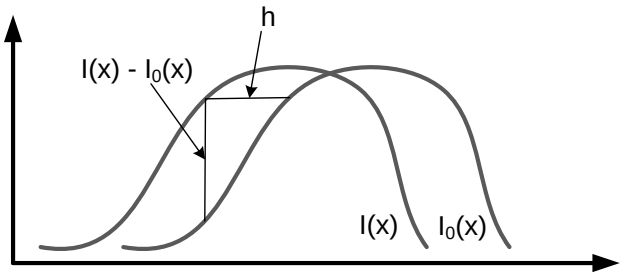


Figure 3.3: Registration of two images.

in motion. The other way round, not all intensity changes can be traced back to motion, but also to changes in illumination or noise. Despite this issue, it is assumed that the intensity or brightness of one pixel does not change when moving from one frame to the next, which is expressed by the *brightness constancy* constraint [25]:

$$I(\boldsymbol{x}, t+1) = I(\boldsymbol{x} - \boldsymbol{u}, t). \tag{3.10}$$

By forming a first-order Taylor-Expansion of Equation (3.10), the general *optical flow constraint*:

$$[I_x \; I_y]\boldsymbol{u}^T + I_t = 0 \tag{3.11}$$

assumes that the gradients in both directions, displaced by the optical flow vector $\boldsymbol{u} = [u \; v]$ and the temporal direction, sum up to zero. Basically, it can be differentiated between between *local* and *global* methods. In addition, combined methods generate a dense optical flow field using robust techniques derived from local methods, eventually trying to combine the best of both worlds.

Local Optical Flow Estimation

The local optical flow approach assumes that the optical flow within a windowed neighborhood is constant. The original approach has been introduced by Lucas et al. in 1981 [39] and was initially proposed for registration of stereo images. The algorithm is based on a spatial image-gradient.

The problem of aligning two images is solved by minimizing the corresponding pixel intensity differences. This is accomplished by applying a distance measure such as the L^2 norm.

Based on the brightness constancy constraint in Equation 3.10, the image is displaced by a certain amount $\boldsymbol{u} = [u \; v]$ between the two points in time t to $t+1$. Figure 3.3 depicts an one-dimensional example. The function $I_0(x)$ is shifted by the amount of the displacement vector h towards the function $I(x)$. Using this geometrical relations and the known values of $I_0(x)$ and $I(x)$, the displacement vector h can be computed by:

$$I(x+h) = I(x) + hI'(x), \tag{3.12}$$

where $I'(x)$ is the first-order derivative of $I(x)$. Minimizing the displacement h with respect to Equation 3.12 will lead to the derivation of the energy function:

$$\frac{\partial E}{\partial h} = \frac{\partial}{\partial h} \sum_x [I(x) + hI'(x) - I_0(x)]^2 \tag{3.13}$$

$$\frac{\partial E}{\partial h} = \sum_x 2 * I'(x)[I(x) + hI'(x) - I_0(x)] \tag{3.14}$$

Data: images $I(x)$,$I_0(x)$,$I'(x)$
Result: displacement vector h
$h_0 = 0$;
$k = 0$;
$\epsilon = \infty$;
while $\epsilon > \epsilon_{min}$ *and* $k < k_{max}$ **do**
  $h_{k+1} = h_k + (\sum_x w(x)I'(x+h_k)[I_0(x) - I(x+h_k)])/(\sum_x w(x)I'(x+h_k)^2)$;
  $\epsilon = |I_0(x) - I(x+h_k)|$;
  $k = k+1$;
end

Algorithm 3.1: Local Optical Flow Computation [39].

Reformulating Equation 3.14 will will lead to an iterative formula to compute the value of h iteratively:

$$h = \frac{\sum\limits_x I'(x)[I_0(x) - I(x)]}{\sum\limits_x I'(x)^2}. \tag{3.15}$$

The naive approach of shifting the displacement vector h across the image leads to a complexity of $O(m^2n^2)$ for an image having an area of $m * n$. To decrease this computational effort, the displacement vector can be computed iteratively starting from an initial vector h_0 (Algorithm 3.1). The weight vector $w(x)$ is given by $1/(I'_0(x) - I'(x))$. In every step, its difference to the value of $I_0(x)$ is computed. The algorithm will stop after a minimum convergence threshold ϵ_{min} or a maximum amount of iterations k_{max} has been reached.

Considering the discrete image space, the iterative update can be implemented by replacing the differentiation operator using e.g. forward differencing:

$$I'(x) \approx \frac{I(x + \Delta x) - I(x)}{\Delta x}. \tag{3.16}$$

This method can be extended to an affine registration problem by enhancing the displacement vector with a deformation matrix. Although this would increase the flexibility of the model, it is encouraged to use this method only to correct the tracking process between the first and the current frame and use the simpler translational model for inter-frame tracking [54]. Another extension to the local optical flow estimation algorithm concerns the use of multi-resolution images to tackle the problem of features at different scales. The displacement vector is computed initially at the highest pyramidal level L_m and propagated upwards until level L_0. This is done by integrating over the error function E with a constant search window $(2\omega_x + 1) \times (2\omega_y + 1)$ for all pyramidal levels [7]. Because in the initial contribution, only gray-level changes are considered, this method can also be extended to incorporate all available color channels. The implication of this change is that a multiple times more constraints are available for solving the optical flow constraint in the search window, i.e. one set of constraints per color channel [1].

Global Optical Flow Estimation

One major disadvantage of estimating optical flow only in a local window is that homogeneous regions, that means regions which have similar color, have inappropriate conditions towards the set of equations from which the local optical flow vector $[u\ v]^T$ should be estimated. A solution to this problem is given by a different approach, which tries to compute the optical flow field for the whole image instead of considering only local neighborhoods. Mathematically, a second constraint $s(\boldsymbol{u})$, the *smoothness term* is necessary to determine a

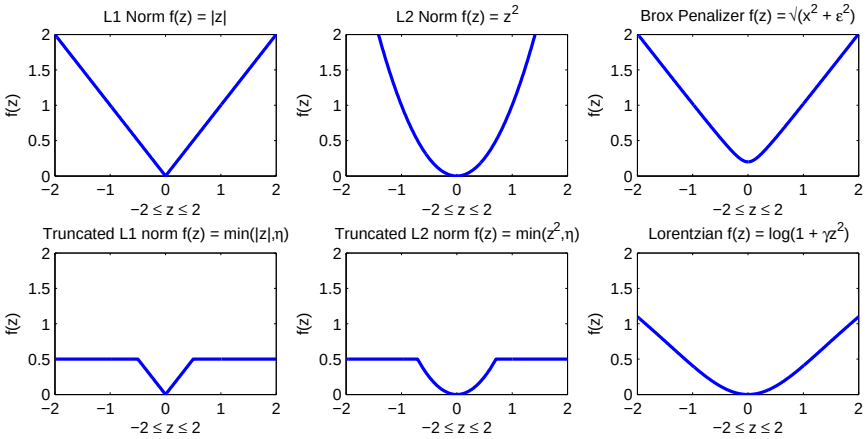


Figure 3.4: Different error functions $\Psi(x)$.

unique motion field:

$$s(\boldsymbol{u}) = (\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2, \tag{3.17}$$

which is the magnitude of the gradient of the optical flow velocity in both x and y direction. While local methods allow neighboring optical flow vectors to possibly contradict each other, this smoothness constraint allows adjacent vectors to differ only to a certain degree. The error functional:

$$E(u,v) = \int\int \Psi(\alpha s([u\ v]^T)) + I_x u + I_y v + I_t) dxdy, \tag{3.18}$$

combines the smoothness term $s(\boldsymbol{u})$ with the data term given by Equation 3.11. The value of α determines the influence of the smoothness term. To minimize this error function incrementally, discrete approximations for the Laplacian and partial derivatives must be given. The partial derivatives in x,y, and t direction are estimated by averaging over the first four differences over neighboring pixels in the cube representing the image sequence. The discretization for the Laplacian is given by a discrete Laplacian filter mask. The iterative Gauss-Seidel method is used to solve the error functional [25]. If a quadratic error function such as $\Psi(x) = x^2$ is used, the proposed function is not robust against potential outliers. In order to address this problem, other robust error functionals replace the quadratic penalizer. Figure 3.4 depicts different error functions which can be used for optical flow estimation. The L^1 norm would be a robust alternative, however, because the corresponding Euler-Lagrange derivation is non-trivial, a robust approximation to the L^1 penalizer is given by $\Psi(x) = \sqrt{x^2 + \epsilon^2}$ [9].

Combined Local and Global Optical Flow Estimation

Combined local and global methods try to estimate a dense motion field while at the same time use the data term from local methods. This has the advantage of computing a dense motion field while being robust to noise at the same time [10]. The combined functional can be described by:

$$E_{CLG}(u,v) = \int\int ([\ u \quad v \quad 1\ ]\, K_\rho * (\nabla_3 I \nabla_3 I^T) \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} + \alpha|\nabla w|^2) dxdy. \tag{3.19}$$

The standard deviation ρ of the integration kernel K_ρ serves as integration scale and defines the influence of the neighborhood. The larger the value of $\rho > 0$ the more is the method robust against noise. The vector

$\nabla_3 I$ holds the partial derivatives in x,y and t direction. In homogeneous regions, no local information can be computed because the corresponding eigenvalue matrix:

$$\begin{pmatrix} K_\rho * (I_x^2) & K_\rho * (I_x I_y) \\ K_\rho * (I_x I_y) & K_\rho * (I_y^2) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -K_\rho * (I_x I_t) \\ -K_\rho * (I_y I_t) \end{pmatrix} \tag{3.20}$$

is not invertible. However, the combined approach is now able to fill in optical flow information using the smoothness term. A multi-grid approach can further improve the accuracy of the presented method. This approach computes the optical flow field at the coarsest level and refines it iteratively through finer levels. At each level $l < l_{max}$, a residual error $r^h = I^h - A^h \tilde{x}^h$ is computed, where h is the current grid size, I^h is defined as $1/\alpha(K_\rho * I_x I_t, K * \rho * I_y I_t)$ and $\tilde{x}_h$ is the concatenated vector $\begin{bmatrix} u^h & v^h \end{bmatrix}^T$. This residual error is corrected at level $l+1$ to make the algorithm more efficient. The matrix A_h defines the corresponding entries of the previous recursive estimation step [10].

3.2 Appearance Representation

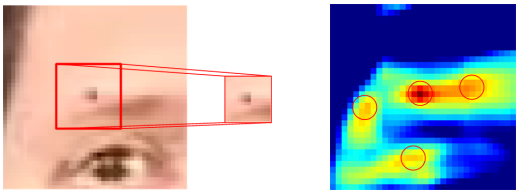
The appearance representation defines how the selected features are arranged to form a model of the tracked object. Usually, even if multiple similar objects should be tracked, the selected representation concerning each target's appearance may be specific to a single object and does not generalize to all objects of the same kind. The appearance representation is also associated with a metric which determines the score or likelihood of a candidate matching the original model.

3.2.1 Template

One way to represent the appearance of the target is using a template (Figure 3.5a) which encodes the pixels' intensities together with their positions in the target area. A score function is used to evaluate each image position for its match with the extracted template. An example for a score function is the *sum of squared differences* (SSD) error already presented in section 3.1.3, which is a L^2 distance based error metric:

$$d(\boldsymbol{x}) = \sum_{\boldsymbol{u} \in I_T} (I(W(\boldsymbol{u}, \boldsymbol{x})) - I_T(\boldsymbol{u}))^2, \tag{3.21}$$

where $W(\boldsymbol{u}, \boldsymbol{x})$ describes a transformation function to map the coordinates of the template I_T to the coordinates of the candidate image I. Because both the squared image $I(\cdots)^2$ and and the squared template $I_T(\cdots)^2$ stay



(a) Extracted Template and Search Region

(b) Cross Correlation Result with Local Maxima

Figure 3.5: Extraction of template and resulting correlation map. Images best seen in color.

constant when expanding this error function, only the cross correlation term :

$$c(\boldsymbol{x}) = \sum_{\boldsymbol{u} \in I_T} (I(W(\boldsymbol{u}, \boldsymbol{x}))I_T(\boldsymbol{u})) \tag{3.22}$$

influences the error function. One problem with the naive implementation of cross correlation is its error-proneness to intensity changes. This means, instead of the naive cross correlation approach, the normalized cross correlation is used:

$$\frac{1}{|I| - 1} \sum_{\boldsymbol{u} \in I_T} \frac{(I(W(\boldsymbol{u}, \boldsymbol{x}) - \overline{I})(I_T(\boldsymbol{u}) - \overline{I}_T)}{\sigma_I \sigma_{I_T}}, \tag{3.23}$$

where $\overline{I}$ and $\overline{I}_T$ represent the mean images of the candidate and template region, respectively. Figure 3.5 shows an extracted template and the resulting cross correlation image. The center in Figure 3.5b shows the maximum of the normalized cross correlation response image, which corresponds to the true location of the marker. The red circles indicate all other maximal values of the correlation image based on the normalized cross correlation function. In addition to a similarity function, a search strategy is necessary. Apart from a correlation-based search strategy - which could be either global or local -, gradient-based strategies exist [39]. One major disadvantage of templates is their sensitivity to noise and out-of-plane rotations. Additionally occlusions might result in an immediate decline of the associated score function. On the other hand, templates are easy to implement and fast to compute, which can be advantageous in environments with lots of similar objects which appearance hardly changes [41, pp. 76-78].

3.2.2 Histogram

A histogram is a discrete estimate of the distribution of intensity values within an image region. The advantage of a histogram is its invariance to scaling and rotation. Additionally, it is robust to partial occlusions, reduces data and can be computed very efficiently. The reason for these properties is given by the global information which is encoded in the histogram. Its major disadvantage is that it does not encode any spatial information. Additionally, color histograms can be misled by changes in scene illumination, out-of-plane rotations and by background clutter. The histogram is computed by encoding the pixel values into a defined number of *bins*:

$$r_{k,j}(x) = C_h \sum_{i=1}^{n_h} \kappa(|| \frac{x - w_i}{h}^2 ||)\delta[b(I_k, w_i) - j], \tag{3.24}$$

where $\kappa(\cdots)$ defines the kernel profile of bandwidth h and C_k is a normalization factor to ensure that the histogram values add up to 1. The function $b(I_k, w_i)$ associates the pixel at position w_i from image I_k with a histogram bin. The function $\delta(.)$ ensures that only the bin j is selected. The set of pixels of the kernel profile is defined by the set $\{w_i\}_{i=1}^{n^h}$. The kernel profile depends on the selected kernel. The simplest kernel is the unit kernel, which is defined by:

$$\kappa(x) = \begin{cases} 1 & \text{if } \|x\| \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

The unit kernel is a hyper-sphere with radius 1 around the origin. Other kernels are based on more complex profiles such as the Epanechikov kernel:

$$\kappa(x) = \begin{cases} 1 - x & \text{if } \|x\| \leq 1, \\ 0 & \text{else.} \end{cases} \tag{3.25}$$

or the Gaussian kernel:

$$\kappa(x) = \exp(-\frac{1}{2}x) \quad x \geq 0. \tag{3.26}$$

The Epanechikov kernel is special because it minimizes the *mean integrated squared error* although the choice of the kernel is not that important compared to the selection of the bandwidth h [13]. Having the histograms, the *Bhattacharyya* coefficient [6]:

$$\rho(r, r_\mu) = \sum_{j=1}^{N_b} \sqrt{r_j r_{\mathcal{M}_j}}, \tag{3.27}$$

where r_j and $r_{\mathcal{M}_j}$ are the corresponding candidate and model values of bin j, can be used to define a distance measure to compare two histograms with the same number of bins N_b:

$$d(r_k(x), r_{\mathcal{M}}) = \sqrt{1 - \rho[r_k(x), r_{\mathcal{M}}]}, \tag{3.28}$$

with $r_k(x)$ being the candidate histogram of image I_k at position x and $r_{\mathcal{M}}$ being the model histogram. In case of color images, a histogram can be encoded in different dimensions. Because histograms basically represent a probability distribution, the joint distribution of the three color channels of RGB or HSV images has three dimensions and for example $8 \times 8 \times 8$ bins. A different option is to marginalize the joint distribution and concatenate the three separate distributions of each color channel to three one dimensional histograms, each having 8 bins. The advantage of the joint histogram is the encoding of the correlation among the three color channels, which is apparently lost when only marginal histograms are considered. On the other hand, the marginalization allows the different channels to be weighted differently, which is important in cases where e.g. lighting changes might effect the *Value* channel of HSV images.

3.3 Localization

Until now, the features have been defined and the representation of the model has been chosen. The next step in a tracking scheme concerns defining the method of estimating the position in the current frame, only given the position in the previous frame and a model of the corresponding object. The presented methods additionally make use of the representation's associated scoring method to rank candidates at different positions according to their suitability to represent the target.

3.3.1 Single Hypothesis Localization

Kalman Filter

As described in Section 4.3, the Kalman filter can also be used for target tracking. It provides an optimal solution under the assumptions of linearity for both the system model and the observation model and the necessity that the posterior distribution $p_{t-1}(x_{t-1}|y_{1:t-1})$ is Gaussian distributed. As it will be explained in Chapter 4, this distribution is part of the recursive Bayes tracking method and is able to predict a state x_{t-1} by considering all the previous measurements $y_{1:t-1}$ from time 1 to $t-1$. If only linearity holds the Kalman filter provides a reasonable solution if the underlying distribution can described fairly well by its mean and variance[41, pp. 96-98].

KLT Tracking

A different approach of tracking uses a combination of feature points proposed by Shi et al. [54] with local optical flow tracking as described in Section 3.1.3. Instead of a shape, the target is described by a set of corner points. These corner points are initially selected by evaluating their robustness using the gradient matrix G:

$$G = \sum_{x=p_x-\omega_x}^{x=p_x+\omega_x} \sum_{y=p_y-\omega_y}^{y=p_y+\omega_y} \begin{bmatrix} I_x^2(x,y) & I_x(x,y)I_y(x,y,) \\ I_x(x,y)I_y(x,y) & I_y^2(x,y) \end{bmatrix}. \tag{3.29}$$

The selection of the corner points is based on the properties of the corresponding eigenvalues of the gradient matrix G in the corresponding search window $(2\omega_x + 1) \times (2\omega_y + 1)$. The eigenvalues λ_1, λ_2 of Matrix 3.29 identify the point as being in an corner, edge or homogeneous region. This means, based on the eigenvalues, the quality of the pixel towards tracking can be expressed. Good feature points meet the criteria:

$$\min(\lambda_1, \lambda_2) > \lambda, \tag{3.30}$$

where λ is a threshold for the noise level [54]. An earlier publication [22] describes the so called *Harris* features. It also uses the eigenvalues λ_1 and λ_2 from the gradient matrix G (Equation 3.29) but applies a different criteria to determine good feature points:

$$\lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 > \lambda, \tag{3.31}$$

with k being an empirically determined constant.

After selection of corner points in the desired ROI, the feature points are located in the next frame $t+1$ by using the optical flow information from the previous frame at time t by simply applying the flow vectors u and v at the current positions of the feature points.

If the shape is known a priori and enough feature points are located on the contour of the target, a shape model can be fitted using RANSAC [17] or Least-Squares-based approaches [18]. Although KLT tracking itself is not able to deal with occlusions, different strategies are possible to compensate for lost features. Each feature is weighted by its consistency, which would be low for features compensated during occlusion situations. Multiple non-interacting targets can be tracked by running multiple KLT trackers in parallel, however if targets may interact or even pass by each other, trackers might confuse the feature points, especially if a compensation strategy is used.

Mean Shift

Mean Shift originated from the field of kernel density estimation which is also known as Parzen window technique. The major advantage of kernel density estimation is that it does not assume a specific form of distribution. Kernel density estimation is non-parametric technique, which means that there are no parameters to describe the distribution. A continuous function is estimated from discrete observations of the feature space. This way, the mode of a distribution can be found without knowing its shape. Mean Shift is not limited to tracking but can be also applied to image segmentation or edge detection [11] as well.

A set of N values $\left\{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^d\right\}_{i=1}^{N}$ is used to estimate a d dimensional function $\hat{f}_k$:

$$\hat{f}_k(\boldsymbol{x}) = \frac{1}{Nh^d} \sum_{i=1}^{N} K(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}), \tag{3.32}$$

where the contribution of each value $\boldsymbol{x}_i$ is given by a *kernel function* $K(\boldsymbol{x}_i)$. The value h describes the bandwidth of the kernel and is used for normalization of the search space.

The most interesting kernels for Mean Shift are the ones which are radially symmetric. This means that the kernel $K(x)$ can also be written as:

$$K(x) = c_{k,d}\kappa(||x||^2). \tag{3.33}$$

The function $\kappa(x)$ is now the *profile* of the kernel. The value $c_{k,d}$ is a d dimensional normalization constant.

Using the profile notation and the derivative of Equation 3.32, the *shadow kernel*

$$g(x) = -\kappa'(x) = -\frac{d\kappa(x)}{dx} \tag{3.34}$$

describes the negative gradient of the kernel profile. This gradient is used to compute the *mean shift vector*:

$$\mathbf{m}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g((||\boldsymbol{x} - \boldsymbol{x}_i)/h||^2))}{\sum_{i=1}^{n} g((||\boldsymbol{x} - \boldsymbol{x}_i)/h||^2)} - \boldsymbol{x}. \tag{3.35}$$

The new position is obtained by applying the mean shift vector to the current position $\boldsymbol{x}$. Figure 3.6 shows an example of using Mean Shift in set of two dimensional data points. In every iteration, the mean shift vector is estimated within the current kernel region represented by the black circle. The estimate of the resulting distribution is used to find the location with the maximum density within the kernel window. The algorithm will eventually converge to the region with the greatest density, which is in the center of the example figure.

In order to apply Mean Shift for tracking, a target model which is based on a histogram as described in Section 3.2.2 is used. The histogram estimates a distribution of color values. In every iteration, the target model $\hat{\mathbf{q}} = \{\hat{q}_u\}_{u=1:m}$, where m is the amount of bins, is compared with a candidate histogram $\hat{\mathbf{p}}(\boldsymbol{x}) = \{\hat{p}(\hat{\boldsymbol{x}}_{t-1})_u\}_{u=1:m}$ at position $\boldsymbol{x}$. To compare a candidate model with the target model, the Bhattarchyya coefficient (Equation 3.27) is used. Algorithm 3.2 describes one Mean Shift time step to estimate the position at time t from the previous position at time $t-1$. To ensure convergence, the model is updated until the Bhattacharyya coefficient $\rho\left[\hat{\mathbf{p}}(\hat{\boldsymbol{x}}_t), \hat{\mathbf{q}}\right]$ is greater or equal than the coefficient $\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{x}}_{t-1}), \hat{\mathbf{q}}\right]$ of the previous frame. If this is not the case, an average of the position at $t-1$ and t is computed to prevent the estimate from being trapped in a local minimum.

Although Mean Shift tracking is computationally inexpensive, it uses only a single hypothesis to describe the target distribution. As soon as multi-modal distributions in the form of clutter or other targets occur, it may converge to the wrong location.

Several extensions to Mean Shift have been proposed. One approach incorporates SIFT [38] features into the kernel density estimation algorithm [68]. The backprojected weights (Algorithm 3.2)

$$w_i = \sum_{u=1}^{m} \sqrt{\hat{q}_u / \hat{p}_u(\hat{\boldsymbol{x}}_{t-1})}\, \delta\left[b(y_i) - u\right], \tag{3.36}$$

are reweighted according to the SSD measurement of SIFT features in the ROI of the Mean Shift tracker. This means that not only color information but also SIFT features are considered by creating correspondences from one frame to the next. In addition, instead of the standard Epanechikov kernel, a Gaussian kernel is used, which means that Mean Shift effectively changes into an *Expectation-Maximization* (EM) algorithm. When compared with classical Mean Shift which uses only color histograms, the combined approach that uses SIFT features as well, is able to reduce the error by 44 percent. However, since the computation time for one frame consisting of only one target takes about 1 second, it is unsuitable for the amount of targets required in the scenario used in this thesis.

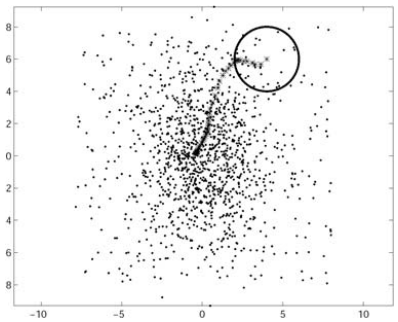


Figure 3.6: Mean Shift kernel and density estimation.

Data: $\{\hat{q}_u\}_{u=1:m}$,$\hat{\mathbf{x}}_{t-1}$
Result: $\hat{\mathbf{x}}_t$
Initialize location with $\hat{\mathbf{x}}_{t-1}$ current frame;
Compute $\{\hat{p}(\hat{\mathbf{x}}_{t-1})_u\}_{u=1:m}$;
Evaluate $\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{x}}_{t-1}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{x}}_{t-1})\hat{q}_u}$;
while $||\hat{\mathbf{x}}_t - \hat{\mathbf{x}}_{t-1}|| > \epsilon$ **do**
    Derive weights according to $w_i = \sum_{u=1}^{m} \sqrt{\hat{q}_u / \hat{p}_u(\hat{\mathbf{x}}_{t-1})}\delta\left[b(y_i) - u\right]$;
    Find next location using 3.35;
    Compute $\{\hat{p}_u(\hat{\mathbf{x}}_t)\}_{u=1:m}$;
    Evaluate $\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{x}}_t), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{x}}_t)\hat{q}_u}$;
    while $\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{x}}_t), \hat{\mathbf{q}}\right]$ ¡ $\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{x}}_{t-1}), \hat{\mathbf{q}}\right]$ **do**
        $\hat{\mathbf{x}}_t = 0.5(\hat{\mathbf{x}}_{t-1} + \hat{\mathbf{x}}_t)$;
    end
end

Algorithm 3.2: Mean Shift algorithm [13].

3.3.2 Multiple Hypothesis Localization

In contrast to single hypothesis methods, multiple hypothesis methods generate multiple hypothesis for each frame. The usage of multiple hypothesis makes these algorithms robust against occlusions and clutter. One popular method is the particle filter algorithm. As described in Chapter 4, every hypothesis - or so called *particle* - is validated against the observed data given by images and features. The influence of unlikely particles is lowered from frame to frame until they are eventually pruned. Despite their robustness towards noise, clutter and occlusions, one major disadvantage of multiple hypothesis localization (MHL) is their computational expensiveness. This is strongly related with the *curse of dimensionality* as the number of samples required to exhaustively explore the state space grows exponentially with its number of dimensions [41, pp. 98-99].

Chapter 4

Non-Linear Bayesian Tracking

Different problems in science require investigation of the unknown state of a system that changes dynamically. The *state-space* approach is able to model a dynamic system and to deal with multi-variate data. The evolution of the system is modeled by differentiating between the hidden state and measurements which arrive at discrete times. A state vector has all relevant information to describe the system under investigation, which, in case of tracking, might just be the object's position or second order variables such as its kinematic properties. This chapter is intended to introduce the concepts around tracking methods relying on this state-space approach. It starts with explaining the methods around discrete Bayes filters (Section 4.1) and describes how this approach can be extended to a continuous state space (Section 4.2). This theory also is fundamental to subsequent sections, which deal with two popular instances of Bayes filters. While the first one has certain restrictions to the probability distribution of the state space (Section 4.3), the other one is able to deal with any posterior probability distribution. This non-analytical approach (Section 4.4) does not rely on single hypothesis, but uses multiple hypothesis to estimate a probability distribution of an object's state vector. Because this approach is also based on *sampling*, different techniques are discussed in this section as well. Throughout this chapter, the notation $p(\dots)$ is used for a probability density and $P(\dots)$ is used for a probability function.

4.1 Hidden Markov Models

In case of discrete states, a *Hidden Markov Model* (HMM) can be employed, which describes a graph where its vertices represent both system states and observations, and its edges represent probabilistic transitions among its vertices. Considering the example in Figure 4.1, the model consists of a set of discrete hidden states $x_i \in \{x_1, ..., x_6\}$ and a set of observations $y_i \in \{y_1, ..., y_4\}$. Every transition from one state x_i to another state (solid) and every transition from a state to an observation y_i (dashed) is associated with a probability $P(y_i|x_i)$. This probability is used in conjunction with the *Bayes-Theorem* (Equation 4.1) to compute probabilities along a path within a HMM. The Bayes theorem, which is named after his originator Thomas Bayes, started out as a solution towards *inverse probability* and deals with a set of theorems treating probability as a result of partial trust towards an event, instead of the event's frequency.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.1}$$

Once an HMM is instantiated, a Markov process is created with a prior probability $P(x_0)$, emitting a (noisy) observation at every time step $1, 2, ..., t-1, t$ (Figure 4.2). The transitions from one state to the next are modeled as conditional probability $P(x_t|x_{t-1})$. Additionally, the probability of the measurement which is emitted by the state at time t is given by the conditional probability $P(y_t|x_t)$. This conditional probability is also called the *likelihood* of the observation to be emitted by the corresponding state.

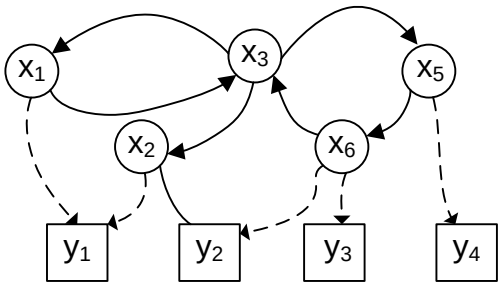


Figure 4.1: Hidden Markov Model with discrete states.

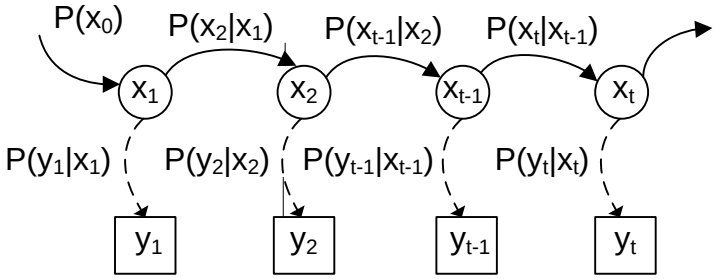


Figure 4.2: Instance of Markov process.

4.2 Bayes Filter

Indeed, this distinction of observed measurements and unknown states applies to many problems in computer vision, such as the problem of tracking objects. In this case, instead of a set of discrete states, a continuous state-space has to be considered. The unknown state is often associated with location and velocity of an object, while the observed information only consists of positional information potentially being clutter. This means that the dimension of the measurements is usually lower than the state which is estimated. Signal acquisition, such as camera calibration, and environmental influences, such as lighting conditions, often perturb the measurements with noise. The state-space approach is able to deal with multivariate and non-Gaussian/non-linear data as well. These properties of the state-space approach render advantageous when compared with similar methods which try to estimate an unknown state [63]. With every step, the current state x_{t-1} is advanced to the next point in time t and a new measurement y_t is drawn from the current state x_t (Figure 4.2). In order to compute the state of the dynamic system, at least two different models are necessary:

- A **system model**, which describes the transition from one state to the next:

$$\mathbf{x}_t = f_t(\mathbf{x}_{t-1}, \mathbf{v}_{t-1}), \tag{4.2}$$

  where $f_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_x}$ is a (in general non-linear) transition function. In this case, n_x and n_v are the dimensions of the state vector and the noise vector, respectively. Applying this function on the state $\mathbf{x}_{t-1}$ at time $t-1$ advances it to the next state $\mathbf{x}_t$, while adding system noise $\mathbf{v}_{t-1}$.

- An **observation model**, which describes how an observation is derived from the current state:

$$\mathbf{y}_t = h_t(\mathbf{x}_t, \mathbf{w}_t), \tag{4.3}$$

  where $h_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_y}$ describes the derivation function to model the transition from the current state x_t to the observed measurement y_t. Similar to the state transition, measurement noise $\mathbf{w}_t$ is added

to the observed state based on the measurement. The variables n_w and n_y are the dimensions of the measurement noise vector and the measurement vector, respectively.

These two models can be formulated probabilistically to find an estimate of the current state x_t by incorporating all measurements $y_{1:t}$ up to time t. This probabilistic formulation makes the Bayes approach suitable because the observations can be seen as events updating the current knowledge by using the Bayes theorem (Equation 4.1). The complete problem can now be modeled as a posterior probability density function easily [53], allowing to obtain state estimates from the distribution. An additional requirement towards tracking is that observations are not available at once, but only one at a time, similar to the Markov process described in Figure 4.2. To fulfill these requirements, the Bayes filter approach divides the described process into two different steps: *prediction* and *update*. Before these steps are outlined in detail, two additional simplifications are necessary. To ensure that the probability of the state transition to the state x_t only depends on its previous state x_{t-1}, the Markov property is necessary for the probabilistic model:

$$P(x_t|x_{t-1:0}) = P(x_t|x_{t-1}). \tag{4.4}$$

Additionally, it is required that the likelihood of the current measurement y_t only depends on the current state x_t, hence, the observations are independent from each other:

$$P(y_t|x_{t:0}) = P(y_t|x_t). \tag{4.5}$$

4.2.1 Prediction

If the required probability density function at time $t-1$ is available, the prediction step uses the system model (Equation 4.2) to generate a prior probability density function for the current time t:

$$p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}, \tag{4.6}$$

where $p(x_t|x_{t-1})$ is the probabilistic form of the system model (Equation 4.2). The prediction step spreads and deforms the probability density function because of the uncertainty introduced with the state transition function [53].

4.2.2 Update

The second step incorporates the observation y_t at time t into the dynamic system:

$$p(x_t|y_{1:t}) = \frac{P(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t|y_{1:t-1})}. \tag{4.7}$$

where the normalizing constant

$$p(y_t|y_{1:t}) = \int P(y_t|x_t)p(x_t|y_{1:t-1}dx \tag{4.8}$$

depends on the likelihood function $P(y_t|x_t)$ which is given by the observation model (Equation 4.3).

In general this problem is intractable and cannot be computed analytically. However, given some restrictions to the system and observation model [53], approaches such as the Kalman filter (Section 4.3) exist, which are able to provide an analytical solution. A general approach which is able to provide a solution without these restrictions, is Monte Carlo tracking (Section 4.4).

4.3 Kalman Filter

The Kalman Filter has been introduced by Rudolf E. Kalman in 1960 [29]. Originally, it was used for removing noise in measurement devices. Compared to the general model introduced in the previous section, the Kalman filter makes two assumptions concerning the tracking environment. First, both the system model function and the observation function are modeled as linear function as given in Equations 4.9 and 4.10.

$$x_t = F_t x_{t-1} + v_{t-1} \tag{4.9}$$

$$y_t = h_t x_t + n_t \tag{4.10}$$

Secondly, optimality is only guaranteed if the estimated location in the state space is Gaussian distributed, hence $p(x_t|y_{1:t}) \approx N(x_t; \mu_t, \Sigma_t)$. In this environment, the Kalman filter provides an optimal solution [53]. Consider the computation of a simple dynamic system for which the average should be computed. Because observations are not available at once, the estimated average has to be updated whenever a new measurement is added. The basic idea is that the estimate

$$\hat{x}_t = \frac{1}{t}\sum_{i=1}^{t} y_i = \frac{1}{t}\sum_{i=1}^{t-1} y_i + \frac{1}{t} y_t \tag{4.11}$$

at step t will be directly computed from the estimate $\hat{x}_{t-1}$ of the previous step at time $t-1$:

$$\hat{x}_t = \frac{t-1}{t}\hat{x}_{t-1} + \frac{1}{t} y_t, \tag{4.12}$$

leading to the update step of the Kalman filter:

$$\hat{x}_t = \hat{x}_{t-1} + \frac{1}{t}(y_t - \hat{x}_{t-1}) \tag{4.13}$$

This shows how to reformulate the initial batch problem as recursive problem which can be solved analytically. Each part of Equation 4.13 corresponds to a different part of the Kalman Filter:

- $\hat{x}_t$ describes the state at time t.
- $y_t - \hat{x}_{t-1}$ describes the *innovation*, which is the difference between the expected (estimate) and actual (measurement) value.
- $\frac{1}{t}$ describes the *gain* which specifies to which degree the innovation can be trusted for the current time t.

4.4 Monte Carlo Tracking

The Kalman filter is restricted to provide an optimal solution only if the system is linear and Gaussian (Section 4.3). In situations where any of these constraints are not given, sequential Monte Carlo methods provide a better solution because they are able to model any type of non-linear system and any form of posterior probability density function $p(x_t|y_{1:t-1})$.

Monte Carlo methods rely on the method of *sampling* from a probability density function $X \sim p(x)$ to compute its expected value $\mathbb{E}(X)$. In this section, importance sampling and parallel importance sampling are outlined. Additionally, an example is given which compares the introduced concepts with uniform sampling.

4.4.1 Importance Sampling

Suppose $P(x)$ and $Q(x)$ are two probability distribution functions. In cases where it is difficult to draw samples from $P(x)$, it is not possible to compute $\mathbb{E}(f(x)) = \int f(x)dx$ directly. If the second function $Q(x)$ is proportional to $P(x)$ it can be used generate samples instead. When evaluating these samples given by $Q(x)$ directly the expectation value would be:

$$E_Q(f(x)) = \int Q(x)f(x)dx, \tag{4.14}$$

which is not equal to $\mathbb{E}(f(x))$. Using a scaled result $g(x) = f(x)/Q(x)$ to remove the influence of $Q(x)$ will result in the correct expectation value of $f(x)$:

$$E_Q(g(x)) = \int Q(x)g(x)dx = \int Q(x)f(x)/Q(x). \tag{4.15}$$

The last part $1/Q(x)$ is also called the *importance weight* of the sample x[50].

To generate a Gaussian distributed random number $x \sim N(\mu, \sigma)$ from uniformly distributed random numbers, the Box-Muller transformation [8] can be used. Two uniformly sampled numbers u_1 and u_2 are necessary which can in turn be used to generate two numbers $\{\tilde{x}_1, \tilde{x}_2\} \sim N(0, 1)$ by using $\tilde{x}_1 = \sqrt{-2\ln u_1}\cos(2\pi u_2)$ and $\tilde{x}_1 = \sqrt{-2\ln u_1}\sin(2\pi u_2)$. These numbers can be transformed easily to distributions of different mean and standard deviation. In Appendix A a pictographic example is given which compares importance sampling with uniform sampling. Uniform sampling is sampling a random number within an interval $[a, b]$.

4.4.2 Parallel Importance Sampling

In cases we do not know $P(x)$ but $\lambda P(x)$, which means that the original distribution is scaled by some different value λ, the acquired samples from $Q(x)$ do not represent the target function $f(x)$. To estimate the value of λ, the raw weights can be computed:

$$\hat{w}_i = \lambda P(x_i)/Q(x_i). \tag{4.16}$$

The expectation value $\mathbb{E}(\hat{w}_i)$ of these raw values result in an unbiased estimate of λ to correct the posterior distribution function towards the actual importance weight $P(x)/Q(x)$. The problem here is that the variance of this estimator is high, so instead of using $\hat{w}_i$ directly, it is normalized by $\bar{w} = 1/N \sum_i \hat{w}_i$ which is also an unbiased estimate of λ, but with a lower variance [50]. This is also directly associated with the normalization step of the importance weights in *particle filtering* introduced in the next section.

4.4.3 Metropolis-Hastings Sampling

Algorithms which sample from probability distributions by creating a Markov chain are called *Markov chain Monte Carlo* (MCMC) methods. These algorithms are initialized with a starting value and executed until the desired probability distribution reaches an equilibrium. Similar to importance sampling, MCMC methods are able to sample from a probability distribution $P(x)$, requiring only that a function $Q(x) \propto P(x)$ is available. A special type of MCMC algorithms are those which are based on Gaussian random walks. That means, the creation of new samples is a Gaussian process and at the same time bound to an accept-reject ratio which is determined randomly. Metropolis-Hastings [23] is one of these random-walk algorithms. The general idea is that each sample will be either accepted or rejected, based on the ratio between the probability of the new sample $\acute{x}$ and the current sample x_t at time t. The major disadvantage of importance sampling methods is that they are not suitable for an increasing amount of dimensions because of exponentially increasing variance of the weights w_i [30]. The Metropolis-Hastings sampling method is summarized in Algorithm 4.1. It uses a proposal - or jumping - density function $Q(\acute{x}|x_t)$, which suggests a new sample value $\acute{x}$ given a sample value x_t (Line 3). Very often, the proposal distribution is symmetric and Gaussian centered at x_t, to explore samples

in its neighborhood region. In subsequent steps, the generated sample is evaluated based on its acceptance ratio a (Line 4 onwards). This ratio determines if the new sample should be rejected or accepted. The first M samples will be regarded as calibration samples - or *burn-in* samples, which means that they do not belong to the final set of samples (Line 7). The major difference to importance sampling is that the generated samples are not independent, but correlated, because the samples are linked by a Markov chain. The larger the amount of samples N, the smaller the error between the real probability function $P(x)$ and the estimated probability $\hat{P}(x)$ given by the set of samples.

Data: Proposal density $Q(x|\acute{x})$, arbitrary point x_0 as first sample, burn-in rate M

Result: set of $N - M$ random numbers $\mathrm{X} = \left\{x_t^{(j)}\right\}_{j=1}^{(N-M)}$

1 $X \leftarrow \emptyset$;
2 **for** $i \leftarrow 1$ **to** N **do**
3 $\quad \acute{x} \leftarrow Q_f(x_t))$;
4 $\quad a \leftarrow P(\acute{x})/P(x_t)$;
5 $\quad$ **if** $a \geq 1$ **then** $x_{t+1} \leftarrow \acute{x}$;
6 $\quad$ **else** Accept $\acute{x}$ with probability a;
7 $\quad$ **if** $i > M$ **then**
8 $\quad\quad X = X \cup x_{t+1}$;
9 $\quad$ **end**
10 **end**

Algorithm 4.1: Metropolis-Hastings MCMC algorithm [23].

4.5 Particle Filtering

Sequential Bayes filtering, or *particle filtering* [53], considers a weighted set of samples (Set 4.17) to represent the posterior probability density $p(x_t|y_{1:t-1})$.

$$\left\{x_t^{(j)}, w_t^{(j)}\right\}_{j=1}^{N}. \tag{4.17}$$

The weighted samples are generated using parallel importance sampling (Section 4.4.2). The importance distribution $Q(x)$ is replaced by the state transition function, which is given by the system model $p(x_t|x_{t-1})$. In more complex cases, other importance functions can be used to incorporate contextual information. The target function $f(x)$ for which the expected value $\mathbb{E}(f(x))$ should be computed, is replaced by the posterior probability density function $p(x_{0:t}|y_{1:t})$. The function $P(x)$ simplifies to the likelihood, which is given by the observation model (Equation 4.3). To iteratively compute new estimates, particle filtering uses *Sequential Importance Sampling* (SIS), which combines both concepts of importance sampling (Section 4.4.1) and the two-stage state estimation of the Bayes filter (Sections 4.2.1 and 4.2.2). The estimate of the current state is given by the weighted average of the existing set of particles (Equation 4.18). This ensures that outliers (i.e. samples which represent a state which is prone to model clutter) do not influence the estimate.

$$\hat{x} = \frac{1}{N}\sum_{j=1}^{N} w_t^{(j)} x_t^{(j)} \tag{4.18}$$

Since the weight $w_t^{(j)}$ at time t is computed from the previous weight $w_{t-1}^{(j)}$ at time $t-1$, only one small set of samples might have a high weight while the others are of negligible importance after some time. This is called the *degeneracy problem* and occurs if the set of samples does not change. To prevent these situations, instead of only sampling from the same set of samples, a *resampling* stage needs to be introduced, which uses weighted

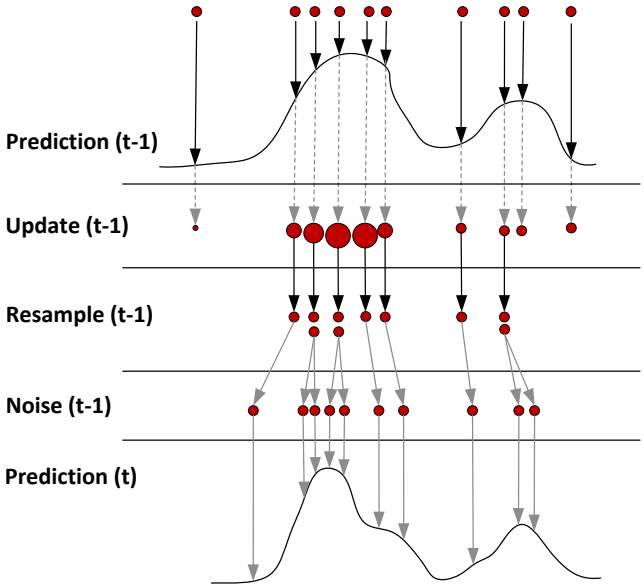


Figure 4.3: Estimation of posterior density function $p(x_{0:t}|y_{1:t})$ from $t-1$ to t.

randomized sampling to generate a new set of samples based on the existing set of of samples

$$(x_{t-1}^{(j)}, w_{t-1}^{(j)})_{j=1}^{N}. \tag{4.19}$$

However, because samples with low weight are more likely to be removed, the state space is now more poorly covered with possible hypothesis. This leads to the problem of *sample impoverishment*. To resolve the problem of balancing these contradicting effects, one has to be careful about when resampling should be performed. The most popular criteria is the effective sample size $\widehat{N_{eff}}$ given by:

$$\widehat{N_{eff}} = \sum_{j=1}^{N} (w_t^{(j)} w_t^{(j)})^{-1}, \tag{4.20}$$

which needs to be smaller than a defined threshold N_s to induce resampling.

Algorithm 4.2 gives a pseudo-code description of the presented algorithm. Additionally, Figure 4.3 outlines the presented scheme using a graphical example. From the available probability distribution at time $t-1$, each sample is weighted against the available measurement. Using a uniform distribution, a new sample set is generated according to the current sample weights. Finally, Gaussian noise is added, resulting a in a predicted probability distribution for time t. If in the last step of adding the process noise $\mathbf{v}_{t-1}$, Gaussian noise is used, the corresponding noise vector has to be correlated according to the covariance matrix of the process. The *Cholesky decomposition* [48, pp. 994] $\Sigma = LL^T$ is used to decompose the covariance matrix Σ into its lower triangle L. When this matrix is applied to the noise vector $\mathbf{v}_{t-1}$, a sample vector $L\mathbf{v}_{t-1}$ is produced, which has the necessary covariance properties of the process model.

A combination of Mean Shift (Section 3.3.1) with sequential Bayes estimation techniques such as particle filtering allows to combine a deterministic with a stochastic component. One approach [40] is using Mean Shift to find the local modes of the distribution after the weighting step of the particle filter algorithm. Although this increases the accuracy of the estimated positions, it would be computationally intensive to apply Mean Shift to

every single particle in the set. Instead, Mean Shift is usually applied on the target estimate $\hat{x}$ instead.

Data: $\left\{x_{t-1}^{(j)}, w_{t-1}^{(j)}\right\}_{j=1}^{N}, y_t$

Result: $\left\{x_t^{(j)}, w_t^{(j)}\right\}_{j=1}^{N}$

foreach *Sample* $(x_{t-1}^{(j)}, w_{t-1}^{(j)}) \in \left\{x_{t-1}^{(j)}, w_{t-1}^{(j)}\right\}_{j=1}^{N}$ **do**
    Draw sample $x_t^{(j)} \sim p(x_t|x_{t-1})$;
    Weight samples according to $w_t^{(j)} = w_{t-1}^{(j)} P(y_t|x_t^{(j)})$;
end
Normalize importance weights s.t. $\sum_{j=1}^{N} w_t^{(j)} = 1$;
Update weighted estimate $\hat{x} = (\sum_{j=1}^{N} w_t^{(j)} x_t^{(j)})/N$;
if $\widehat{N_{eff}} < N_s$ **then**
    Resample;
end

Algorithm 4.2: Sequence Importance Sampling with Resampling [53].

Chapter 5

Interactive Tracking of Markers

In this chapter, the implemented and evaluated methods are described in detail. In the subsequent sections, the different steps executed in the image processing pipeline are explained step by step. This explanation starts with the preprocessing step explained in Section 5.1. The steps concerning marker localization are presented in Section 5.2. As multiple targets are being tracked, explanations of the proposed strategy using data association is given in Section 5.4. Finally, the interaction of the operator with the tracking system is described in Section 5.5.

5.1 Preprocessing

Before the targets are tracked using different localization methods, two preprocessing steps prepare the image for tracking. The image is denoised using a bilateral filter 5.1.1 and the facial region is extracted using segmentation methods 5.1.2. The resulting image is then used for tracking.

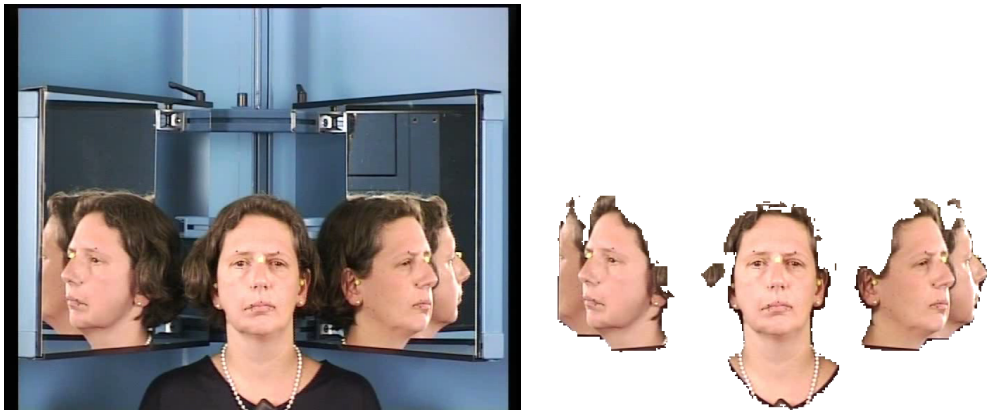
5.1.1 Filtering

The image is filtered using a bilateral filter, because of its edge-preserving property in contrast to a Gaussian filter. The idea is based on a non-linear combination of nearby image pixels in a given local area. Neighborhood pixels which are on the opposite side of an edge have much less influence than pixels lying on the same side of the edge. This is accomplished by adding an additional range term which measures the difference between the center pixel and the neighborhood pixel [59]. Since we use two different camera setups in our experiments, two images of each sequence using either setup was selected as training images. The parameters of kernel size, spatial Gaussian σ_s and color Gaussian σ_c have been optimized by naively iterating over combinations of a set of ranges. For the kernel size a range of $[3, 9]$, for both σ_s and σ_c a range of $[1, 56]$ was evaluated. To evaluate the accuracy of noise removal, the *Peak-Signal-to-Noise Ratio* (PSNR) is computed. The PSNR defines the ratio between the maximal value of the signal and the power of corrupting noise and is computed by

$$20 \lg(\max(I)) - 10 \lg(\frac{1}{mn} \sum_{(i,j) \in I,J} [I(i,j) - J(i,j)]^2), \tag{5.1}$$

where $I(\cdots)$ and $J(\cdots)$ are two monochrome images of size $m \times n$ and Max_I is the maximal value of image $I(\cdots)$.

The parameters with the highest PSNR value are selected for further experiments. For our camera setup (Section 6.1.1), this resulted in an optimal kernel size of 4×4 and σ_c and σ_s being both 1. For the VMU (Vienna Medical University) setup, the same values of standard deviation where chosen, however, the optimal kernel size determined to be 3×3. To evaluate the parameters, 3 images are selected from sequences of either camera setup.



(a) (b)

Figure 5.1: An example image before (a) and after (b) the segmentation has been applied. (Courtesy of VMU)

This results in an average PSNR of 98.00 $\pm$ 0.08 for our camera setup and 89.4408 $\pm$ 5.1879 for the VMU camera setup.

5.1.2 Face Segmentation

To enhance the robustness of the tracking system, it is crucial to distinguish foreground from background and to limit the search space in which the candidate model is searched. Several approaches exist to tell regions having interesting image features and the background scene apart. The requirement in our case is however limited to an approximate segmentation, accurate enough to separate the facial region from the background and any other disturbing image regions. The face segmentation is only enabled for facial sequences and disabled for the evaluation of non-facial sequences.

A thresholding method in the *YCbCr* (Section 3.1.1) space is employed to separate the background from the facial region containing the markers. The face is additionally processed with morphological operators and large holes are closed to ensure that the mask image has only one connected component containing the facial region. Figures 5.1a and 5.1b show an example of images before and after the segmentation, respectively. Algorithm 5.1 outlines all steps in the segmentation process. The values γ_{Cr} and γ_{Blob} define the thresholds for the Cr channel towards the facial region and the minimum size of the objects which are kept in the scene, respectively. Ideally, only the three facial objects remain which correspond to the three facial images of the mirror setup (Figure 5.1b).

5.2 Tracking

This section describes how the methods introduced in Chapter 3 and Chapter 4 are used to create tracking schemes which extract the features described in Section 3.1 to create appearance models (Section 3.2) of facial markers which are subsequently tracked from frame to frame. The section starts with a description of how the particle filtering scheme is instantiated by incorporating different image cues (Subsection 5.2.1). Additionally, an alternative sampling scheme is presented which is based on the theory from Section 4.4.3. Finally, two baseline schemes are introduced (Subsection 5.2.3 and Subsection 5.2.4) which are compared with the particle filtering methods in the evaluation part of this thesis.

Data: RGB or HSV image I
Result: image with extracted facial region $\acute{I}$
$\hat{I} \leftarrow Cr\ channel\ of\ YCbCr(I)$ (Section 3.1);
foreach $(x, y) \in \hat{I}$ **do**
  if $\hat{I}(x, y) > \gamma_{Cr}$ **then**
    $\hat{I}(x, y) = 1$;
  else
    $\hat{I}(x, y) = 0$;
  end
end
Remove blobs smaller than γ_{Blob} pixels;
foreach $(x, y) \in \hat{I}$ **do**
  if $\hat{I}(x, y) = 0$ **then**
    $\acute{I}(x, y) = I(x, y)$;
  else
    $\acute{I}(x, y) = 1$;
  end
end

Algorithm 5.1: Face Segmentation.

5.2.1 Sequential Monte Carlo Tracking

In the first frame, appearance models that describe the landmarks are built from the selected coordinates given by the user. The operator also has to provide the size of the markers in pixels. From these positions, the target models $\{A_k\}_{k=1}^{T}$ are built. For simplicity, random variables and their realizations have the same notation throughout this section. As described in Section 4.5, The posterior distribution density $p(x_{t-1}|y_{0:t-1})$ at time $t-1$ is given by a set of samples:

$$\left\{x_{t-1}^{(j)}, w_{t-1}^{(j)}\right\}_{j=1}^{N}. \tag{5.2}$$

An initial set of samples:

$$\left\{x_{0}^{(j)}, w_{0}^{(j)}\right\}_{j=1}^{N} \tag{5.3}$$

is generated from the initial coordinates given by the user. Based on the initial coordinates of the markers, the particles are separated into different clusters. Thus, the different modes of the posterior distribution density $p(x_{t-1}|y_{0:t-1})$ are separated. Each particle now belongs to a single cluster. Due to this partitioning of the state space using clustering, multiple targets can be tracked using a single particle filter algorithm [36]. After this initial clustering step, the markers are tracked using SIS with resampling (Section 4.5).

For state transition, the first order motion model:

$$x_t^{(j)} = M_t^{(i)} x_t^{(j)} + w_t \tag{5.4}$$

is used, with M_t being the state transition matrix and $w_t \sim N(0, \sigma)$ representing the acceleration part, which is the change of velocity over time. Alternatively, it would be possible to learn the motion prior M_t and provide a more accurate representation of facial movement [3]. Each measurement y_k of the available observations $\{y_k\}_{k=1}^{T}$ is associated with a specific target which in turn is associated (Section 5.4) with a cluster of particles. Instead

of a single observation likelihood, it has been suggested in [41, p. 119] to combine different measurements by fusing them according to a set of weights, one for each likelihood model:

$$P_t(y_k|x_t^{(j)}) = \sum_i \beta_i \acute{P}_t(y_k|x_t^{(j)}), \tag{5.5}$$

with $\sum_i \beta_i = 1$. Each part $\acute{P}_t(y_k|x_t^{(j)})$ of the combined observation model is in turn modeled as Gaussian process with $N(0, \sigma)$:

$$\acute{P}_t(y_k|x_t^{(j)}) = 1/(\sqrt{2\pi}\sigma) e^{-\psi(x_t^{(j)}, y_k)^2 \frac{1}{2*\sigma^2}} + \epsilon, \tag{5.6}$$

where both σ and $\psi(\cdots)$ depend on the chosen appearance model and feature set. The elevation ϵ is necessary to prevent zero likelihood and numerically instable numbers in the implementation. The selection of σ is very important concerning the sensitivity towards clutter. If this parameter is chosen too large, the kurtosis of the corresponding Gaussian density function becomes too low, which means that the difference between the likelihood of designated target and background noise is small. This will also effect the distribution of two sample clusters interfering at close proximity, since the likelihood values start to merge and the boundary between the two likelihood distribution vanishes.

Systematic resampling is used within each cluster k separately, according to the effective sample size $1/\sum_{i=1}^{N_k} |w_i|^2$. It is important to employ measures to stabilize the particle filter algorithm [31] to avoid the problems of sample impoverishment and degeneracy:

- minimize resampling steps, e.g. by a proper selection of N_{eff}
- generate new samples to fill up impoverished samples

This means that the selection of the threshold N_s for the effective sample size $\widehat{N_eff}$ is very crucial.

In the next paragraphs, the different observation models evaluated in this thesis are presented.

Histogram-based Likelihood

The state

$$x_t = \begin{bmatrix} x & \dot{x} & y & \dot{y} & H_x & H_y & \theta \end{bmatrix}^T, \tag{5.7}$$

is modeled as an ellipse which is rotated in a certain angle, where H_x and H_y are the elliptic axis and θ is the rotation. In a different approach [43] a scale variable is used, which is replaced with the angle variable to reflect changes in the elliptic region of the marker. This was chosen because it was assumed that during tracking, the changes of rotation are greater than the changes of scale.

This means that in addition to the two-dimensional coordinates and their corresponding velocities, the extent of the ellipse in both dimensions as well as its rotation angle is reflected in the estimation of the state for the sample $X_t(j)$ at time t. The Bhattacharyya coefficient $\rho[p_s(i), q]$ [6] is used to calculate the congruency between the target and candidate histograms of each sample. An Epanechnikov kernel is used to weight the corresponding values of the histogram. Instead of the original function [43], the distance function $\psi(\cdots)$ is modified to represent a proper distance function bounded with $[0, 1]$ [12]:

$$\psi(x) = \sqrt{1 - \rho[p_s(i), q]}. \tag{5.8}$$

The matrix given by Equation 5.9 describes the state transition function M_t for the color-based likelihood

state, while the matrix in Equation 5.10 describes the associated process noise w_t.

$$M_t = \begin{bmatrix} 1 & \Delta & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Delta_{Hx} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Delta_{Hy} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.9}$$

$$w_t = \begin{bmatrix} \frac{1}{3}\Delta\sigma_x^3 & \frac{1}{2}\Delta\sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}\Delta\sigma_x^2 & \Delta\sigma_x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3}\Delta\sigma_y^3 & \frac{1}{2}\Delta\sigma_y^2 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2}\Delta\sigma_y^2 & \Delta\sigma_y & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{Hx}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{Hy}^2 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_\theta^2 \end{bmatrix} \tag{5.10}$$

Template-based Likelihood

Based on the template appearance model (see Section 3.2.1), the template-based likelihood uses the appearance template given by the state of the sample to evaluate their compliance with a reference target model. The state consists of entries for the position and velocity in both directions:

$$x_t^{(j)} = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^T \tag{5.11}$$

The appearance model is based on the intensity templates and LBP to incorporate texture information. Both information is combined to a fused likelihood image $L_k = (1-\beta)L_{color} + \beta L_{LBP}$. The likelihood image L_k is computed in a search region surrounding the position of the landmark in the previous frame by applying normalized cross correlation over the current frame. The distance function $\psi(x)$ is given by:

$$\psi(x) = 1 - (L_k(W(x)) + 1)/2, \tag{5.12}$$

where $W(x)$ warps the image coordinates into likelihood image coordinates by translation. The appearance model for the target k is updated if the likelihood of the estimated position exceeds a threshold U_γ using:

$$A_k = (1-\gamma)A_k + \gamma\tilde{A}_k, \tag{5.13}$$

where $\tilde{A}_k$ is the candidate model of the current estimate of target k.

Since the acceleration is modeled as white noise, the two parameters σ_x and σ_y control the variance of changes in velocity as the time proceeds from t to $t+1$. The suggested white noise model is given by the matrix in Equation 5.14.

$$w_t = \begin{bmatrix} \frac{1}{3}\Delta\sigma_x^3 & \frac{1}{2}\Delta\sigma_x^2 & 0 & 0 \\ \frac{1}{2}\Delta\sigma_x^2 & \Delta\sigma_x & 0 & 0 \\ 0 & 0 & \frac{1}{3}\Delta\sigma_y^3 & \frac{1}{2}\Delta\sigma_y^2 \\ 0 & 0 & \frac{1}{2}\Delta\sigma_y^2 & \Delta\sigma_y \end{bmatrix} \tag{5.14}$$

$$M_t = \begin{bmatrix} 1 & \Delta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.15}$$

Motion Likelihood and Sampling

The combination of optical flow with the velocity parts of the state vector yields a likelihood metric which is based on motion cue. The approach is based on the cosine similarity between two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$:

$$\cos(\phi) = \frac{\boldsymbol{v}_1 \cdot \boldsymbol{v}_2}{\|\boldsymbol{v}_1\|\|\boldsymbol{v}_2\|}. \tag{5.16}$$

Using cosine similarity, the distance function can be modeled as:

$$\psi(x) = \frac{\tilde{\boldsymbol{x}}_v^{(j)} \cdot N^{-1} \sum_{i \in \mathcal{N}[u\ v]^T}}{\|\tilde{\boldsymbol{x}}_v^{(j)}\| \|N^{-1} \sum_{i \in \mathcal{N}} [u\ v]^T\|}, \tag{5.17}$$

where the average optical flow vector $N^{-1} \sum_{i \in \mathcal{N}} [u\ v]^T$ in the neighborhood $\mathcal{N}$ of sample $\tilde{\boldsymbol{x}}_v^{(j)}$ is compared with its velocity vector.

Since a Combined-Local-Global method (see Section 3.1.3) is used, the average of the motion vector can be computed without removing possible outliers. In other cases, a method such as RANSAC [17] would be necessary to remove outlier optical flow vectors. The main idea behind this approach is that the more congruent the velocity of a sample with the actual optical flow, the more likely is the movement of the sample. Especially in cases where two targets with similar appearance interact, the resulting optical flow field can be used to prevent samples from being hijacked by the wrong target. A similar approach [19] uses a KLT tracker to accomplish the same by defining an energy functional E_m which is based on the difference between the state of the sample $\tilde{\boldsymbol{x}}_v^{(j)}$ and its estimated deterministic position. The estimated position is given by the median of the set of optical flow vectors between the previous frame and the current frame in the region described by the sample.

A different approach of using optical flow is to augment the probability density function for state transition $p(x_t|x_{t-1})$ with the available optical flow information, and create a semi-stochastic system model, which then becomes:

$$f(X_{t-1}^{(j)}) = M_t^{(i)} X_{t-1}^{(i)} + V(X_{t-1}^{(j)}) + w_t, \tag{5.18}$$

where the stochastic sampling process is replaced with a semi-deterministic sampling process based on the optical flow field $V(\boldsymbol{x})$. In this case, the optical flow field determines the velocity from time $t-1$ to time t [5, p. 716].

5.2.2 MCMC Particle Filter with MRF Model

In the previous approach, interactions are not modeled explicitly. Instead, if two targets get to close to each other, one target is probably hijacked by another cluster of samples. The implication of this event is that from that point in time onwards one target is not tracked while the other one is represented by two clusters. Because effectively one target is lost in this case, this approach tries to tackle this problem by using a motion prior $\psi(X_{it}, X_{jt})$ to penalize samples which would result in an overlap of two targets. According to the approach by Khan et al. [30], this constraint can be simply treated as an additional factor in the importance weight. This results in the following state transition density which generates the joint state X_t at time t from the joint state X_{t-1} at time $t-1$:

$$P(X_t|X_{t-1}) \propto \prod_i P(X_{it}, X_{i(t-1)}) \prod_{ij \in E} \psi(X_{it}, X_{jt}), \tag{5.19}$$

Instead of a single target state, $P(X_t|X_{t-1})$ now describes the joint state transition function which models all targets at the same time. The term $P(X_{it}|X_{i(t-1)})$ specifies the single target state transition function, which advances each target $X_{i(t-1)}$ separately, resulting in corresponding states X_{it}. The motion prior $\psi(X_{it}, X_{jt})$ can also be treated as interaction potential of a neighborhood clique containing all targets which are at close proximity to each other. The interaction potential depends on the probability of the neighborhood clique of the corresponding target. A neighborhood clique is formed by considering a Markov-Random Field (MRF), which is a graph (V, E) with undirected edges E between vertices V [34]. Each vertex has a label from the set of

possible labels D. Labels can be either discrete or continuous. A set of random variables $F_i \in F$ assigns a label to each vertex. A realization of the joint event given by the joint set F of random variables is also called the configuration f of the MRF. The potential function is applied on so called *cliques*. A clique describes a local neighborhood which - in the simplest case - consists only of a pair of adjacent vertices. A set of random variables F is said to be a MRF if and only if the following two conditions are satisfied:

1. $P(F = f) > 0 \qquad \forall f \in S$
2. $P(F_i = f_i | F_j = f_j, j \in V, j \neq i) = P(F_i = f_i | F_j = f_j, j \in N_i)$

The first one describes that all probabilities at each vertex must be positive, and the second one describes that the probability of a transition is only allowed to be dependent on the neighborhood clique, which is the Markov property, equivalent to Equation 4.4. The edges represent the joint probabilities as a product of local potential functions ψ. The potential functions $\psi(.)$ for neighborhood pairs consist of two parameters, one for each vertex. A Gibbs distribution is used to define a value based on the transition probability between two vertices $v_i, v_j \in V$ [34]:

$$\psi(v_i, v_j) = \exp(-g(v_i, v_j)). \tag{5.20}$$

In the presented approach, the vertices v_i and v_j are represented by two targets X_{it} and X_{jt}:

$$\psi(X_{it}, X_{jt}) = \exp(-g(X_{it}, X_{jt})). \tag{5.21}$$

The penalty function $g(\cdots)$ depends on the overlap area of the two bounding boxes of each target. It is maximal if the two targets take up the same position and declines as targets move apart.

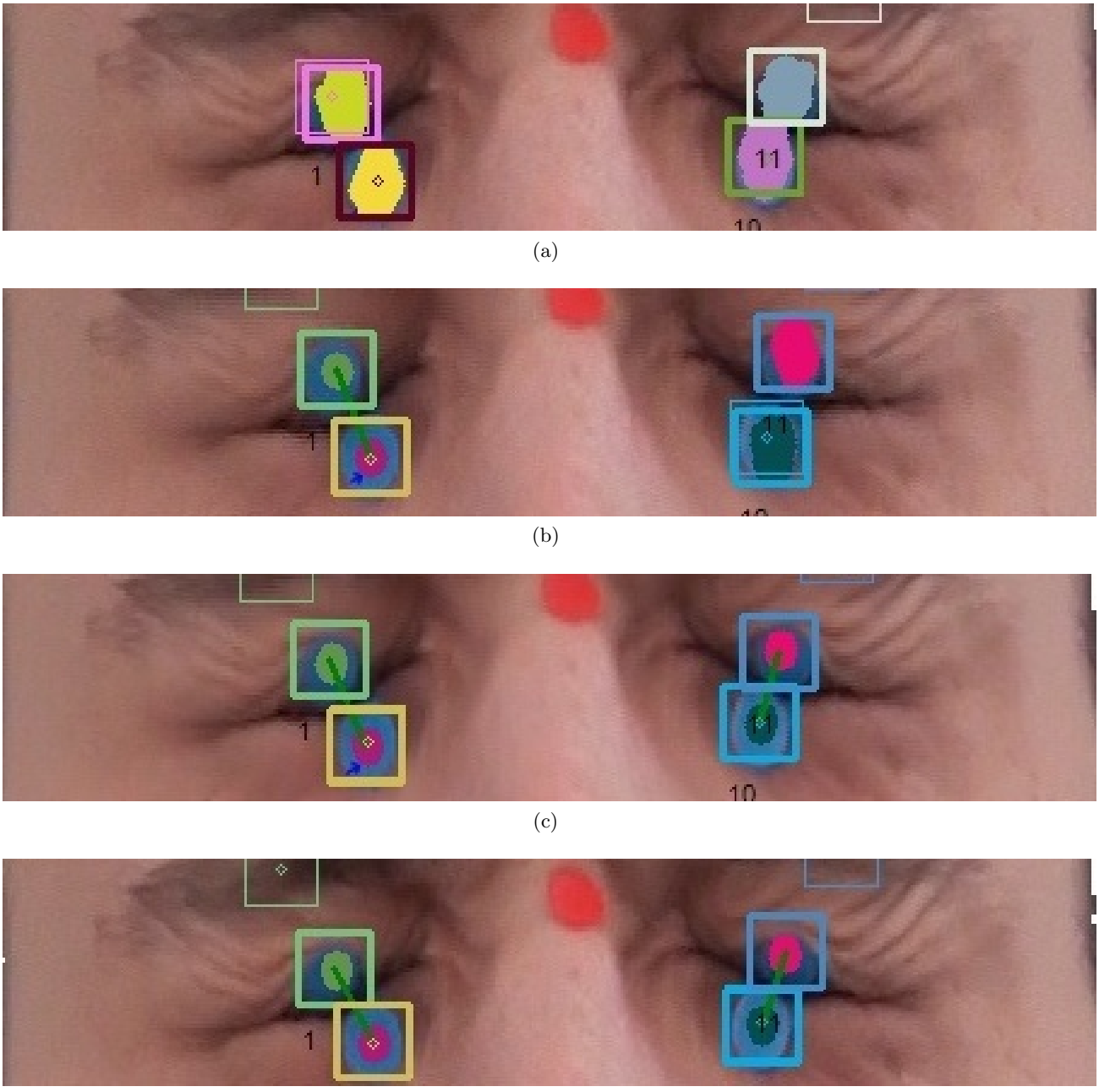
Importance sampling is not able to cope with the higher amount of dimensions when multiple targets are tracked. Considering the color-based particle filter (Section 5.2.1), which has 7 dimensions in its state space, the number of joint target dimensions can be as high as 28, assuming that 4 markers interact with each other. Additionally, it would become more difficult to cover the state space without increasing the amount of samples, as discussed in Chapter 4. This problem is solved by replacing the SIS algorithm with an MCMC (Section 4.4.3) approach. This means that whenever two or more targets interact, all their associated samples are additionally used to advance the joint target while considering the MRF constraint. Metropolis-Hastings (Algorithm 4.1) is then used to generate a set of unweighted samples. The general ratio $\acute{P}(x)/P(x)$ of the algorithm is replaced with an acceptance ratio depending on both the likelihood of the sample $P(Y_t|X_{it})$ and the interaction potential $\psi(\cdots)$ of the MRF graph:

$$a_s = \min(1, \frac{P(Y_t|\acute{X}_{it}) \prod_{j \in E_i} \psi(\acute{X}_{it}, \acute{X}_{jt})}{P(Y_t|X_{it}) \prod_{j \in E_i} \psi(X_{it}, X_{jt})}). \tag{5.22}$$

The initial estimate is generated by joining the estimates of the single targets. From this initial estimate, Metropolis-Hastings is executed and the resulting set of unweighted samples is used to create new samples for each target in the joint target. Every new sample receives the weight $1/N_s$, where N_s is the number of samples in the cluster. A comparison of the different concepts is given in Figure 5.2, which depicts one sample image of the eye region with markers placed on the eye lid. The top-most image (Figure 5.2a) shows that all targets are tracked separately. It uses a particle filter with targets being tracked separately using importance sampling only (Figure 5.2a). The subsequent two figures depict the two left-most (Figure 5.2b and right-most (Figure 5.2c) targets being tracked using MCMC sampling. This means, joint samples are generated for the joint target consisting of two markers each. Finally, Figure 5.2d) shows two cliques of markers being tracked jointly.

Markov Random Fields Likelihood

The penalty Function 5.21 from the previous section can be also used as penalizing likelihood for the particle filter using SIS. In this case, the overlap of the corresponding sample is computed over the graph of all targets



(a)

(b)

(c)

(d)

Figure 5.2: (a): all targets tracked separately, (b)-(d): markers tracked using combined MCMC sampling.

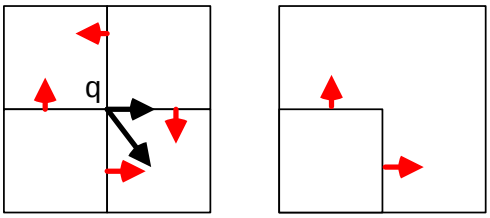


Figure 5.3: Subpixel accuracy by gradient optimization.

with the samples' associated target being compared with all other targets.

$$P_t(Y_k|X_t^{(i)}) = \sum_i \beta_i \acute{P}_t(Y_k|X_t^{(i)}), \tag{5.23}$$

with the likelihood function $\acute{P}_t(Y_k|X_t^{(i)})$ for the overlap defined as:

$$\acute{P}_t(Y_k|X_t^{(i)}) = \prod_{j \in E_i} \psi(X_{it}, X_{jt}). \tag{5.24}$$

5.2.3 KLT Tracking

To compare the Bayes tracker implementation with a baseline approach an extended KLT tracker has been implemented which is able to deal with multiple targets. For each region given by marker location and size, features are determined using the approach described in Section 3.3.1. The feature points are optimized using a sub-pixel aware algorithm before the KLT iteration starts. The iterative optimization of each feature towards its sub-pixel accurate location is based on the observation that every dot product between the image gradient at p_i and the difference vector $q - p$ is zero. That is, if p_i lies within a homogeneous region, the image gradient is zero or if p_i lies on an edge, the gradient vector of the edge is orthogonal to $q - p$, which results in the the dot product being zero (Figure 5.3). For each point p_i of a set of points around q, the residual ϵ_i between the true sub-pixel location and the original center q is defined as:

$$\epsilon_i = [I_x(p_i)\ I_y(p_i)]^T (q - p_i), \tag{5.25}$$

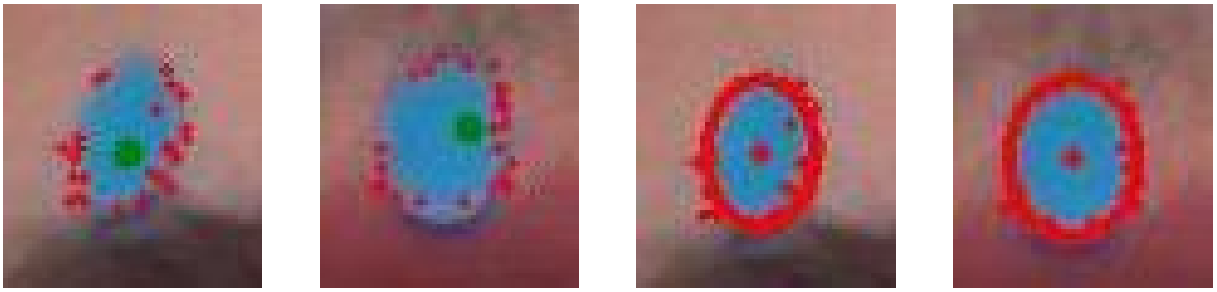
where $I_x(p_i)$ and $I_y(p_i)$ is the gradient at location p_i in x and y direction, respectively. In order to minimize ϵ_i to obtain sub-pixel accuracy, an iterative conjugate gradient method is used to solve the system of equations:

$$q = G^{-1}b, \tag{5.26}$$

which is in turn used to obtain the new center position for the current iteration. Convergence is reached if the center point q is either outside of the search window or the maximum amount of iterations has been reached. The matrix G is the Hessian matrix of the gradients in the neighborhood of the feature points and has the same elements as Matrix 3.29 in Section 3.3.1. The estimate of the marker position is given by the centroid of its available features. Two strategies are explored, the average vector of features and a least squares ellipse fitting method [18]. Figure 5.4 compares the two selected methods.

The used ellipse fitting method is based on a least square approach which tries to find coefficients $\boldsymbol{\theta}$ of the implicit second order polynomial describing the elliptic shape:

$$F(\boldsymbol{\theta}, \boldsymbol{x}) = ax^2 + bxy + cy^2 + ex + f. \tag{5.27}$$



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

Figure 5.4: Comparison of average centroid and direct ellipse fit centroid. Images best seen in color.

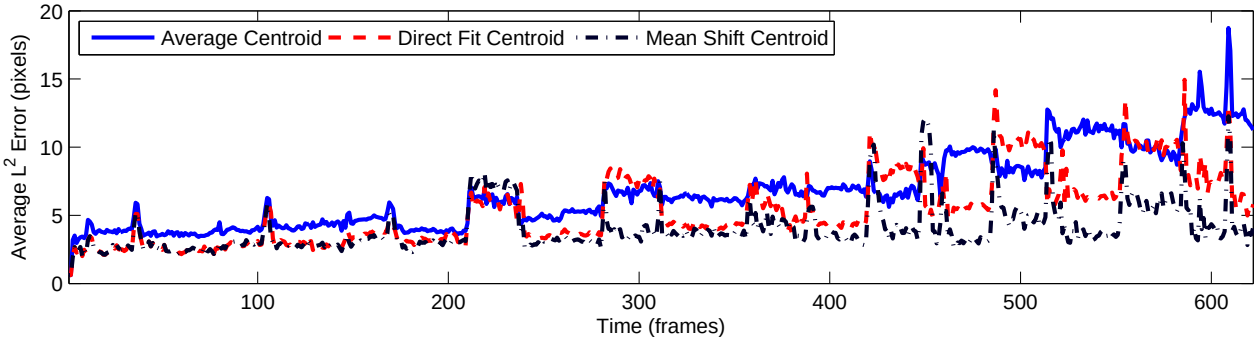


Figure 5.5: Per frame error of the comparison between average centroid (solid line), direct fit centroid (dashed line) and Mean Shift centroid (dash-dotted line), using the *S3* sequence (Section 6.1).

If the least squares approach fails during tracking because the ellipse becomes distorted, the average centroid approach is used as fallback method. Single features are lost if they are outside of the facial region, if the area is too large (i.e. features drift apart), if they are outside of the image boundary, or if a certain amount of features cannot be found with certainty. The whole marker is declared lost and has to be reinitialized manually if any of the features is outside of the facial region after the facial segmentation step. Additionally, a target is declared lost if the area of the estimated ellipse becomes too large. This constraint is based on the observation made during the experiment, that feature points tend to drift apart if their quality becomes worse due to image noise or appearance changes. To compare the different approaches, one video of the institute setup (Section 6.1.1), is evaluated using the different centroid methods. As it can be seen from Figure 5.5, the least squares based direct fit approach increases the accuracy of the estimated position. The least squares ellipse centroid (Figures 5.4c and 5.4d) is able to improve the accuracy compared with the average centroid (Figures 5.4a and 5.4b), because especially when considering bigger markers, its surrounding features are not evenly distributed among the marker contour (Figure 5.4a). During the experiment, it turned out that the ellipse estimated is often misaligned to the marker contour. This is caused by single feature points drifting away from the centroid. Hence, an outlier detection mechanism which uses Mean Shift (Section 3.3.1) computes the Mean Shift center of the feature points. Subsequently, those points where the distance to the Mean Shift center is larger than the marker size can be considered as outliers which are not used for the ellipse fitting method. Figure 5.6 plots the *Average RMSE* of the three different variants. As it can be seen, the Mean Shift correction technique greatly improves the accuracy, especially in the highlighted markers which correspond to the two eye pairs.

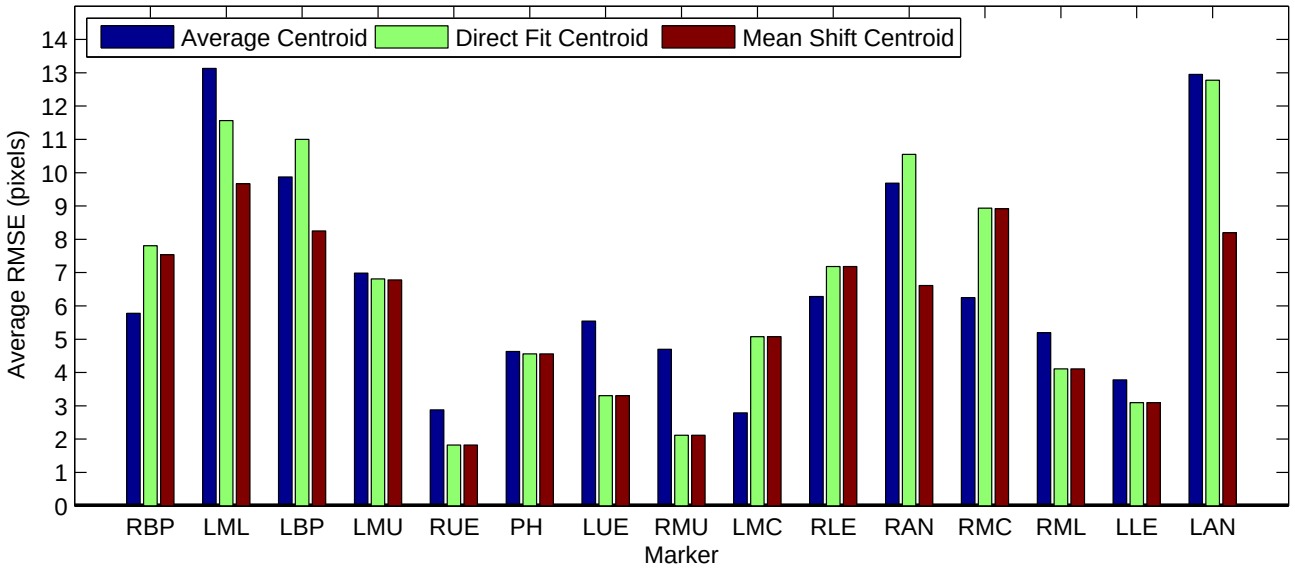


Figure 5.6: Comparison between average centroid, direct fit centroid and Mean Shift centroid per marker.

5.2.4 Mean Shift Tracking

The Mean Shift tracker uses the Mean Shift algorithm as described in Section 3.3.1. For each marker point, a separate tracker is initialized. To ensure that markers do not interfere, Voronoi tesselation (Section 5.3) is used. For every new frame, Algorithm 3.2 is applied until convergence. The resulting position is directly used as new estimate for the corresponding marker location.

5.3 Search Space Separation

The *Voronoi diagram* (Figure 5.7) [2] represents the optimal creation of cells separating a set of sites within a space. Its corresponding dual-graph represents the optimal triangulation among the set of cells. Given a space M and a set S of sites p within M, then the set of all points x which experience strongest *influence* from p belong to the *region* or *receptive field* of p. This property of separating a space into non-overlapping regions is used to ensure that search regions of each targets can be kept separated without the danger of target hijacking. The search space separation is applied using different constraints, which depend on the selected tracking method. For particle filtering, the likelihood was modified to incorporate the restricted search space of the marker:

$$\tilde{P}_t(Y_k|X_t^{(i)}) = P_t(Y_k|X_t^{(i)})sgn(I_k(x,y)), \tag{5.28}$$

where $I_k(x,y)$ is the mask image of the Voronoi tesselation for the target k and $P_t(Y_k|X_t^{(i)})$ is the original likelihood probability. This means that the likelihood is simply set to 0 in regions outside of the restricted Voronoi region. For Mean Shift, the value of the Bhattacharyya coefficient is multiplied with the value of the Voronoi tesselation mask for the given target.

An alternative approach to divide the search space is based on MRF neighborhood constraints. These constraints as described in Section (Section 5.2.2) ensure that targets which are moving to close to each other do

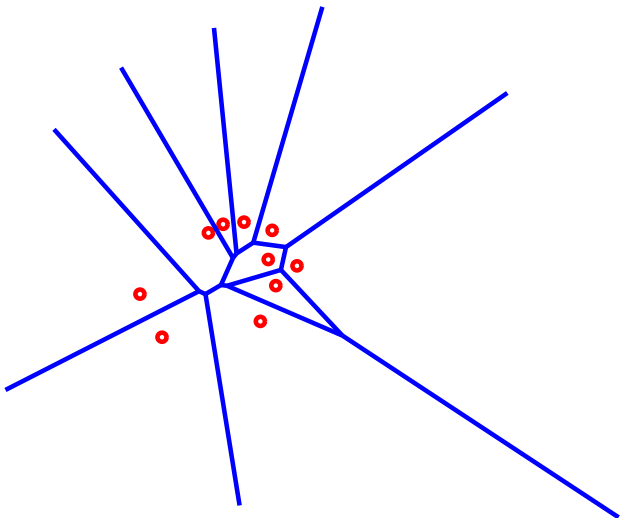


Figure 5.7: Voronoi diagram of random points.

not interfere. The targets represent the nodes V of the MRF graph (V, E). The connections among the targets represent the edges E of the graph. The pairwise interaction potentials

$$\psi(v_i, v_j) = \exp(-g(v_i, v_j)), \tag{5.29}$$

can now be used to impose constraints on the particle filter. The penalty function $g(\cdots)$ is used to separate the targets by adding a constraint on the sampling process, which is based on the overlap area of the two bounding boxes of each target.

5.4 Data Association

Because markers have to keep their identity throughout the tracking process, the unlabeled clusters have to be associated with the labeled targets which represent the facial markers. This makes a data association step necessary, also to allow clusters to be evaluated according to the targets likelihood function $P_t(Y_k|X_t^{(i)})$. A metric is necessary to create a cost matrix on which the association should be based on. Effectively, the Hungarian algorithm [33] searches for optimal associations of a bipartite graph using the costs of its edges. Thus, it yields an optimal assignment between the unlabeled clusters of particles and the labeled targets. The edge cost is defined as the L^2 norm between the estimated position of the cluster of time t and the target of time $t-1$. This results in a unique assignment between clusters and targets.

5.5 Target Reselection

Catastrophic failures, such as target loss or multiple trackers tracking the same target, could not be restored automatically, because there is no object detection method used to reinitialize the tracker. Instead, a semi-automatic approach was chosen which allows the user to reselect a lost or failed target and continue tracking. The malign target will be replaced with a new target generated at the position given by the user. The criteria for target loss is manifold, and distinguishes several types of severe failure:

- The likelihood $P(Y_k, \hat{X}_k)$ of the estimate $\hat{X}_k$ of target k falls below the threshold U_γ.
- The coordinates of the estimate estimate $\hat{X}_k$ of target k are outside of the image bounds.
- The sum of the particle weights of the associated cluster become 0, i.e. after importance sampling, all particles have minimum weight because the target is lost.
- The weighted *absolute average deviation* (AAD_w) exceeds its associated threshold U_{AAD_w}, i.e. the distribution of the samples x and y coordinates is bigger than the marker. The AAD_w metric determines the dispersion of each set of samples which belong to a specific target. It is computed over all N particles participating in a single target estimation:

$$AAD_w = \frac{1}{N} \sum_{j=1}^{N} w_j |x_j - \hat{x}|, \tag{5.30}$$

where w_j is the weight of the sample x_j and $\hat{x}$ is the current estimate of the marker's position.

Chapter 6

Evaluation of Particle Filtering and Baseline Comparison

In this chapter, the methods presented in Chapter 5 are evaluated using different datasets. The datasets differ not only in terms of camera setup but also in terms of difficulty concerning the performed facial expressions. As the goal of this thesis is to evaluate the clinical applicability of object tracking methods towards the presented scenario, different metrics are used to grasp the degree of how these methods cope with the problems listed in Section 1.1. The applicability expresses itself mainly by the degree of how the original time used for manual processing can be reduced. First the method is not allowed to intermix labels of the facial markers, as it will then be impossible to reconstruct the trajectories uniquely. Secondly, it should reduce the tracking time for a single expression while at the same time keep the accuracy compared to manually locating the markers high. As an implying requirement, the methods are also measured concerning their ability to track the sequences with as little interactions as possible.

All experiments have been performed using a non-optimized implementation in .NET, with some parts being written directly in C++, on a dual core Pentium IV with 2.5 GHz and 4 GB of memory. For matrix operations and many other mathematical and image processing functions, Emgu [1] has been used, which is a .NET based wrapper library for OpenCV[2].

6.1 Video Sequences

The presented solution has been evaluated using different video sequences to demonstrate its versatility and at the same time its ability to operate in clinical conditions. In general, the sequences can be subdivided into facial marker sequences and object sequences. Facial marker sequences include one pathological and one non-pathological sequence recorded at the Vienna Medical University and used with their permission. Both sequences have been recorded using setup and workflow as described in Section 2.7. Three additional non-pathological sequences have been recorded using our own camera setup. Different marker colors and sizes have been selected to evaluate the invariance of the presented solutions towards chromatic and spatial properties.

6.1.1 Facial Marker Sequences

The ground truth for facial marker datasets contains manually marked locations of the all markers. At some frames, markers have not been visible during manual selection of the coordinates, even if zooming has been used

[1] http://opencv.willowgarage.com/wiki, Accessed February 27th, 2012.
[2] http://www.emgu.com/wiki/index.php/Main_Page, Accessed February 27th 2012.

during ground truth creation. In case of such image failure, the metrics are not evaluated for the specific frame and marker. The first three subjects (S1, S2 and S3) are recorded using our camera setup, with 1044×1080 resolution. The S5 sequence was recorded using VMU setup with 736×576. The S4 sequence consists of a patient performing a single facial expression with the original head image cut out and the mirror reflections removed (Figure 2.7) with a resolution of 736×576 and a final image size of 179×245. The ground truth contains manually located positions of the all visible facial markers and its resulting trajectories of each marker. All subjects are recorded performing several clinically relevant facial expressions (e.g. smiling) as required by the medical physicians.

The ground truth for all non-pathological cases has been created semi-automatically using a preprocessing stage where an empirically determined color-based threshold for each dataset has been used to segment the regions of interest which contain the markers. Afterwards, in those frames where the automatic segmentation was not able to detect all the markers, they have been located manually. After the positions have been determined in all frames, the Hungarian algorithm has been used to create a trajectory for each observed position. Table 6.1 shows the different non-pathological datasets and their manual processing times. It can be easily seen that as soon as a full sequence has to be tagged in addition to a marker size of several pixels only, tagging the sequences manually takes about 2-3 times longer as seen in the column *Markers per Second*. Because only simple segmentation and morphological operations have been used to ease location of the markers, this largely manual approach can neither be applied to general datasets because they assume a specific color range and cannot cope with markers as small as 3-4 pixels which is the case in the *S4* sequence. Figure 6.1a shows an example frame from the *S5* sequence which uses the VMU setup and Figure 6.1b shows an example frame from the *S3* sequence which uses our camera setup.

The trajectories of the markers are created by creating a cost matrix which contains the L^2 distances among the markers and associating them using the Hungarian algorithm. The last step was to assign a unique label to each trajectory.

6.1.2 Object Sequences

In addition to the application specific ground truth data, general tracking problems of other objects are evaluated and discussed as well. The DARPA Vivid dataset [3] contains some datasets of single objects with different difficulties concerning background, lighting and appearance changes. Ground truth is given by the bounding box of one target in each frame. Depending on the evaluated tracking method, the full bounding box or only its center point is used as prior information to initialize tracking. Groundtruth data is available for one car only. The *egtest01* (Figure 6.2a) sequence consists of a few cars which turn around on a place very similar to their own appearance, then one car is chased by a camera with several fast camera changes. The car accelerates several times and overtakes several other cars during the scene. The *egtest02* (Figure 6.2b) sequence consists of a convoy of cars driving on the border of a multi-lane road. In the middle of the sequence they turn and switch position with a different convoy on the opposite side of the road. Finally, *redteam* (Figure 6.2c) consists of a

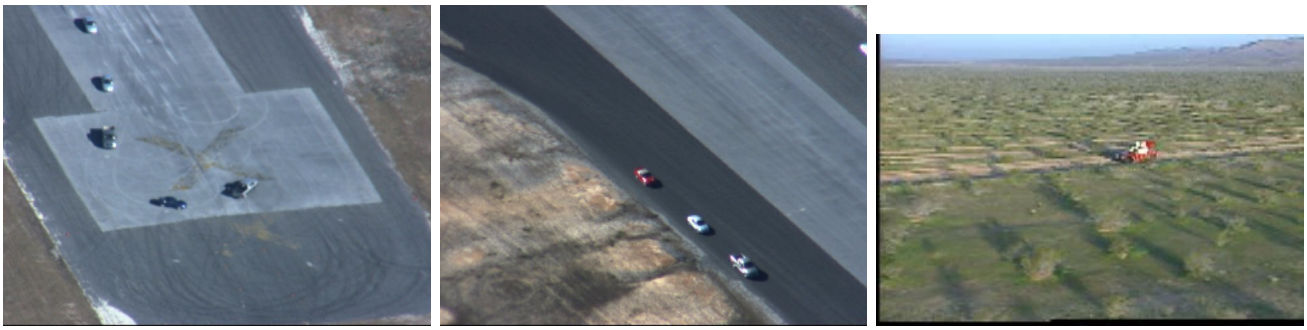
[3] http://vision.cse.psu.edu/data/vividEval/datasets/datasets.html Accessed March 28th, 2012.

Sequence	*# Frames*	*# Markers*	*Time (min)*	*Markers per Second (s)*
S1	623	15	43	0.28
S2	577	15	31	0.19
S3	446	15	41	0.35
S4	176	6	-	-
S5	1510	34	540	0.48

Table 6.1: Overview about manually created ground truth.

(a) (b)

Figure 6.1: Sample images of facial marker sequences: (a) S5 (Courtesy of VMU), (b) S3.



(a) (b) (c)

Figure 6.2: Samples from the DARPA Vivid sequences: (a) egtest01 (b) egtest02 (c) redteam.

single red car moving through a desert landscape. The camera changes pace, scale and angle towards the car very often through the sequence. Table 6.2 summarizes the available sequences.

6.2 Metrics

This section is intended to give an overview on the metrics used to evaluate the different tracking methods.

6.2.1 Euclidean Distance

The Euclidean distance or L^2 distance is computed between the estimated target and the available groundtruth data for each time step t and each target. This value can be kept as is, or averaged over the all targets M_t:

$$\bar{L^2}_t = \frac{1}{|M_t|} \sum_{x_k \in M_t} \|\hat{x}_k - \tilde{x}_k\|, \tag{6.1}$$

with $\hat{x}_k$ being the estimate for target k and $\tilde{x}_k$ being the groundtruth value for target k.

Sequence	*# Frames*	*# Objects*	*Resolution*
egtest01	1820	2-5	640×480
egtest02	1300	3-6	640×480
redteam	1917	1	352×240

Table 6.2: Overview about object sequences from DARPA Vivid.

6.2.2 Root Mean Square Error

The *Root Mean Square Error* (RMSE) measures the accuracy of the tracking system by summing up the residuals between the groundtruth and the estimated positions. The RMSE for the target k is given by:

$$E_{rmse}(k) = \sqrt{E_{mse}(k)} = \sqrt{\frac{\sum_{t=1}^{T} (\tilde{x}_k^{(t)} - \hat{x}_k^{(t)})^2 + (\tilde{y}_k^{(t)} - \hat{y}_k^{(t)})^2}{2T}}, \tag{6.2}$$

with $\left(\tilde{x}_k^{(t)}, \tilde{y}_k^{(t)}\right)$ being the groundtruth coordinate and $\left(\hat{x}_k^{(t)}, \hat{y}_k^{(t)}\right)$ the corresponding estimate at time t for the target k. To measure the total system accuracy, the average value over all targets is computed as well.

6.2.3 Confidence and Likelihood

The confidence measure is defined by the correlation value between the model and the candidate. In order to explore and discuss the ability of the selected features to discriminate the foreground from the background, this measure is recorded along the tracking process. Additionally, the resulting Gaussian likelihood (Equation 6.3) is recorded to investigate the actual behavior of the observation model over time.

$$L_k \sim N(0, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-d(A_k, \tilde{A}_k)^2 \frac{1}{2\sigma^2}}. \tag{6.3}$$

6.2.4 Processing Time

To evaluate the time to process a video sequence, the start and end time of the tracking process is recorded to compute the total time for tracking the sequence in minutes. This value will also allow a comparison with the manual method already in place.

6.2.5 Number of User Interactions

The number of manual interventions is counted throughout the video sequence to grasp a degree of automatism in the corresponding tracking behavior. To make these numbers comparable, the number is normalized towards the number of frames in the video sequence.

6.3 Parameter Selection

The parameters of the methods used in the evaluation are outlined in this section. It starts with the parameters of Mean Shift and KLT tracking (Subsection 6.3.1 and continues with the parameters used in the particle filtering tracking schemes (Subsection 6.3.2).

Variable	*Description*	*Value*
B	Bin size for Mean Shift histogram	8
$\omega_x \times \omega_y$	Window size	15×15
λ	Quality Level	0.2
L	Pyramid Levels	0
M	Number of features per target	100

Table 6.3: Parameters for baseline algorithms.

Variable	*Value*
Δ	0.6
Δ_{Hx}	1.0
Δ_{Hy}	1.0
σ_x	1
σ_y	2
σ_{Hx}	4.0
σ_{Hy}	4.0
σ_Θ	$15.0 * \frac{\pi}{180.0} rad$
$\sigma^{(t)}_{color}$	0.2
$\sigma^{(t)}_{intensity}$	0.2
$\sigma^{(t)}_{lbp}$	0.2
N_{th}	$6.0 * N/10.0f$
ρ_σ	0.2
B_{CPF}	16

Table 6.4: Particle Filter Variables.

6.3.1 Baseline Algorithms

The selected parameters for the baseline algorithms are summarized in Table 6.3. The amount of selected features M for the KLT tracker (Section 5.2.3) is chosen to be 100 per target to ensure that enough features are available for ellipse fitting. The quality level λ affects the actual amount of features as it determines the multiplier for the maximal value which serves as a threshold for the available features in the region of interest. As less features do not impact performance negatively, a relatively low value of 0.2 is chosen, as less high quality features seem to be more important than many low quality features. On the other hand, to less features resulted in problems with the ellipse fitting algorithm, making the selection of this parameter very important. In experiments, it was noticed that when setting the window size $\omega_x \times \omega_y$ to a small window, such as 5×5 instead of 15×15, features drift away and the target is lost after a few frames.

6.3.2 Particle Filter

The parameters of the particle filter are summarized in Table 6.4. They concern the noise of both the system and the observation model (Section 4.2). The values of the likelihood models are used as they are, but they could be also learned using risk minimization [57].

The different σ_* parameters concern the noise which will be applied to every sample during resampling and

Tracking Scheme	*Description*	β_1	β_2	β_3	β_4
TPF	Template-based Particle Filter	0.95	0	0.05	0
Hybrid-TPF	LBP- and Template-based Particle Filter	0.75	0.2	0.05	0
LBP-TPF	LBP-based Particle Filter	0	0.95	0.05	0
CPF	Color-Histogram-based Particle Filter	0	0	0.1	0.9

Table 6.5: Evaluated tracking schemes.

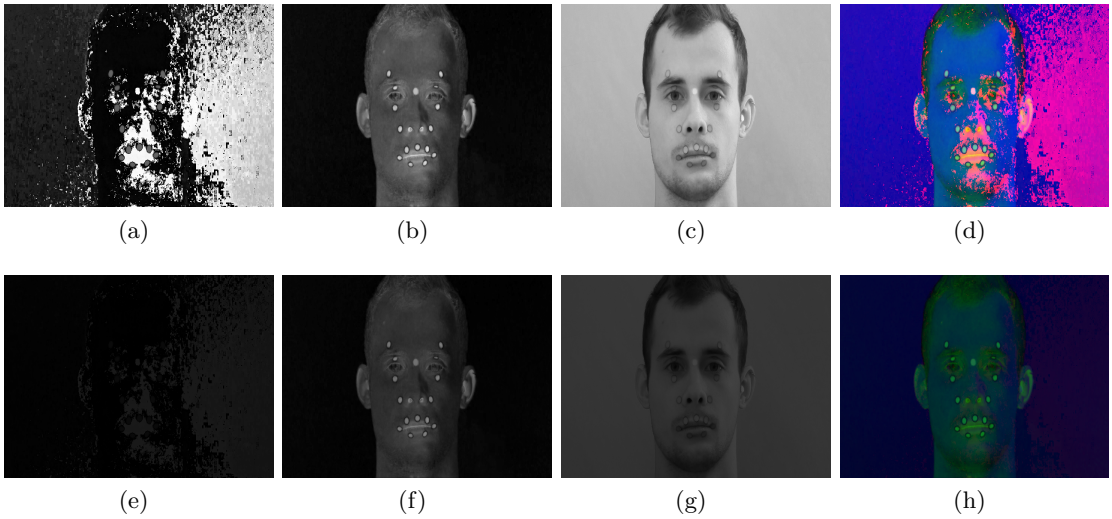
after the state is transitioned from t to $t+1$. The value of σ_y is chosen to be greater than σ_x because more movement on the y (lips, eye region) is expected than on the x axis. The object size is given by the bounding box provided by the corresponding groundtruth of the sequence. Table 6.5 summarizes the different tracking schemes and the corresponding weight parameters for the tracking schemes used throughout this chapter. The variables concerning the likelihood fusion $\beta_{1...4}$ have been chosen empirically, although there are methods to adapt them according to the descriptive power of the corresponding likelihood [41, pp. 123]. Initially, the value of β_3 which concerns the influence of the motion-based likelihood (Section 5.2.1) is set to 0, but an experiment which is described in the following section showed the importance of incorporating deterministic flow information which is why the value is set to 0.05 for subsequent experiments. The variable ρ_σ serves as multiplier for different thresholds, e.g. $\sigma_{color} \times \rho_\sigma$, which determine when the candidate model (histogram or template) will be incorporated into the target model.

6.3.3 Histogram Selection

In the CPF tracking scheme, the target is represented as a weighted marginal histogram of its image region in HSV space. Marginal means that from each color channel a 16-bin histogram is generated, which is then concatenated to form a 3×16 marginal histogram. The values in the histogram are weighted according to the weight vector [0.1 0.6 0.3] to suppress the *Hue* channel, which has a range of $[0°, 25°]$ in facial regions [60]. On the other hand, *Saturation* and *Value* are weighted higher to increase their influence towards tracking. The *Value* channel is weighted half the saturation channel to ensure that lighting changes do not influence tracking. Figure 6.3 compares the original unweighted image with the same image after applying the proposed weight vector. It can be seen that the noise is reduced and the perception of the green markers is increased (Figure 6.3h), compared to the original image in Figure 6.3d. To evaluate the performance between the 3D histogram and the proposed 2D weighted marginal histogram, the sequence *egtest01* was tracked 40 times. While the target was lost 22 times with the 3D histogram, it was lost only 12 times with the weighted marginal histogram. This means that in average, the weighted marginal histogram performs better than the 3D histogram. This weighted marginal histogram is used in all subsequent evaluations.

6.4 Object Sequence Evaluation

In this section, the results of the evaluation using the non-facial sequences *egtest01*, *egtest02* and *redteam* (Section 6.1.2) are presented. The performance of the 4 tracking schemes is investigated both in terms of accuracy under a varying amount of samples (Section 6.4.3), sensitivity to the motion cue parameter (Section 6.4.1) and the variance of results when run repeatedly many times (Section 6.4.2). The percentage of how many frames have been tracked successfully is evaluated through stopping the evaluation when the target leaves the image region. Although both *egtest01* and *egtest02* sequences show multiple moving targets with similar appearance, only one of them is tracked due to limited ground truth. In Section 6.4.3, the four tracking schemes are also compared using two baseline methods, which are Mean Shift (Section 5.2.4) and the KLT Tracker (Section 5.2.3) described in Chapter 5. The *egtest02* sequence has two possible obstacles for the trackers. The first one concerns a convoy



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

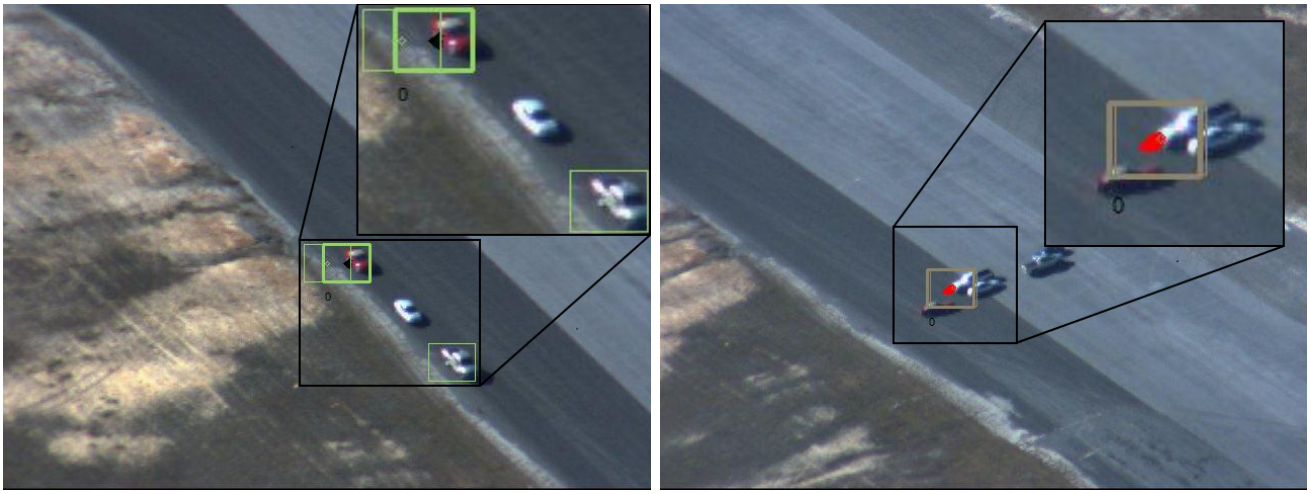
(e) (f) (g) (h)

Figure 6.3: Top row ((a)-(d)) original images, bottom row ((e)-(h).) weighted images. Images best seen in color.

of three subsequent cars with similar appearance. The second obstacle concerns a second convoy passing the convoy which contains the car which should be tracked. Due to these obstacles, the sequence is used to evaluate the motion likelihood (Section 6.4.1).

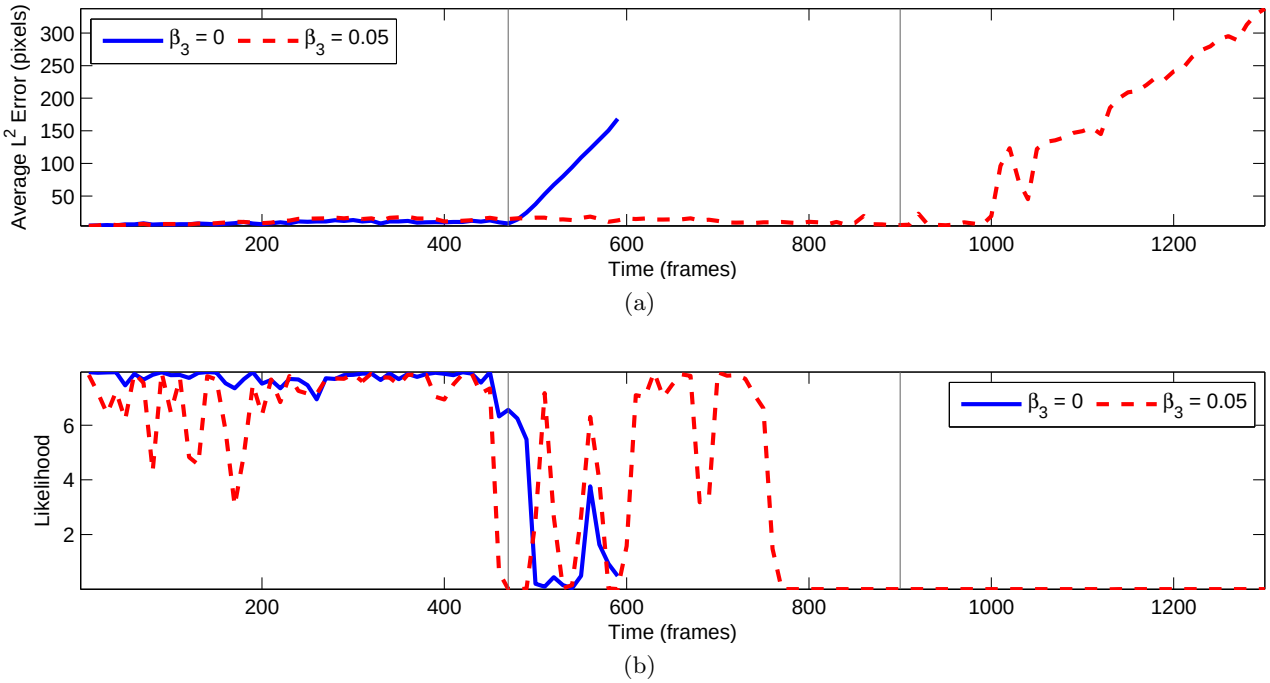
6.4.1 Motion Cue

To analyze the impact of the β_3 parameter, which determines the influence of motion cue in the likelihood of a candidate model, an experiment was conducted. Motion cue consists of incorporation of optical flow information, as described in Section 5.2.1 and can be used to distinguish targets based on their movement. Two values of β_3 are compared on how they influence the ability of the trackers to complete the *egtest02* sequence. The sequence is selected because of its obstacles concerning similar targets. Without considering motion cue, a different car might hijack the tracker or the tracked car might be lost. In case of hijacking, the tracker would stop tracking the correct car and switch to a different one (Figure 6.4b). The TPF tracker starts to track the wrong car at frame 470 due to its similar appearance. By setting β_3 to 0.05, the TPF tracker is not distracted by the arriving convoy and the first obstacle is passed successfully. A comparison of the distance error between a value of 0 and 0.05 for β_3 is given in Figure 6.5a, where the dashed line indicates the increased value. The corresponding likelihood values are shown in Figure 6.5b, and indicate that as soon the two convoys meet (indicated by the first vertical line), the solid likelihood line is higher than the dashed likelihood line. This means that due to the similar appearance the likelihood does not decrease and the wrong target is incorporated into the target model. While in the case of considering motion-based likelihood as well, the likelihood is lower due to a different motion of the two targets, eventually preventing hijacking. When applying the same strategy to CPF, the value of 0.05 does not have enough influence to keep the tracker from tracking the second car. On the other hand, if the value is set too large (0.2 or 0.3) the estimate drifts away from the target because the influence of the motion-based likelihood is too large and the tracker effectively degrades to an optical flow-based tracker (solid and dashed lines in Figure 6.6a). With a value of 0.1 the CPF tracker is able to track the car through both obstacles (dash-dotted line in Figure 6.6a). Even though these changes allowed both trackers to prevent target hijacking, they both lost the target around the second obstacle, indicated by the second vertical line in Figures 6.5 and 6.6.



(a) (b)

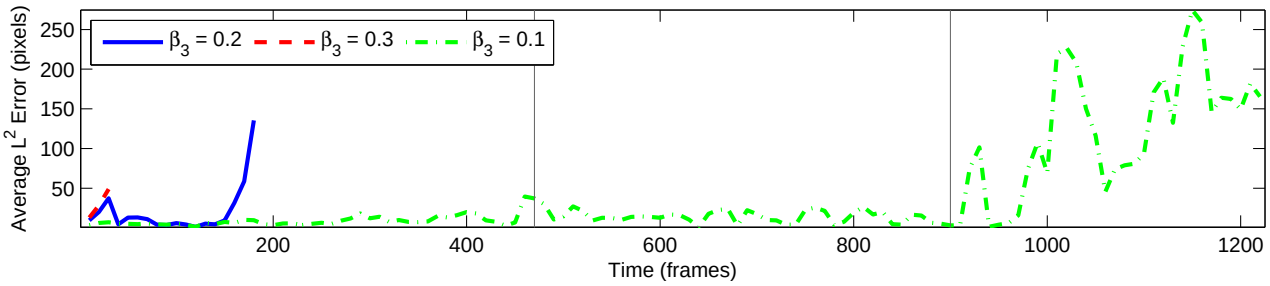
Figure 6.4: (a) LBP and Color template likelihood on different positions (b) Target is hijacked by arriving car of opposite convoy. Image best seen in color.



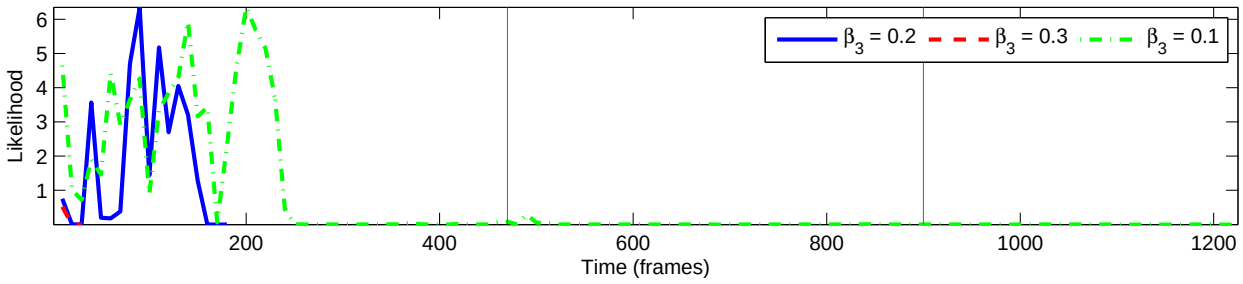
(a)

(b)

Figure 6.5: Comparison of two different values of β_3 for the TPF Tracker applied on sequence *egtest02*.



(a)



(b)

Figure 6.6: Comparison of three different values of β_3 for the CPF Tracker applied on sequence *egtest02*.

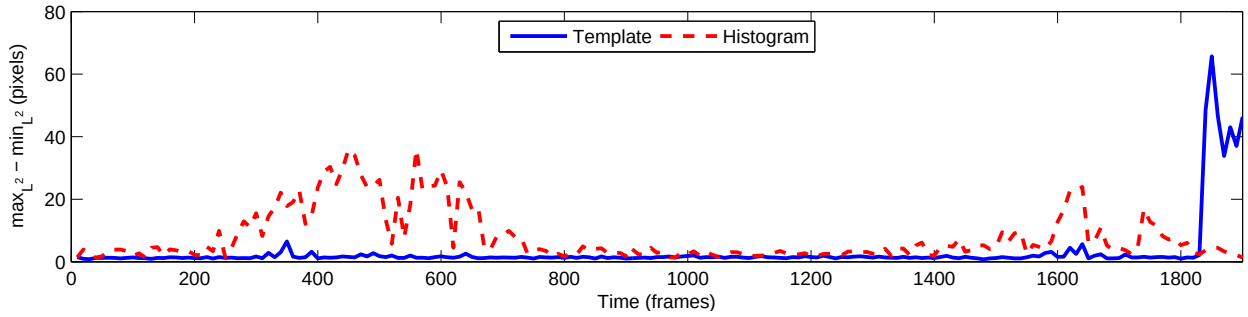


Figure 6.7: System variance of redteam sequence.

6.4.2 System Variance

To investigate the total system variance, each tracking scheme is applied to the *redteam* sequence 50 times. Because all template-based trackers (LBP-TPF, TPF and Hybrid-TPF) have nearly the same result, only a comparison between trackers using a histogram-based likelihood and trackers using a template-based likelihood is given. Figure 6.7 compares the variances of the distance errors for each frame. As it can be seen, in parts of the sequence where the camera changes its pace quickly, the variance of the histogram-based tracker (dashed line) can be 4-5 times higher than the variance of the template-based tracker. On the other hand, in the later part of the sequence where the camera zooms out and induces scale changes of the object, the variance of trackers using a template likelihood (solid line) is higher.

6.4.3 Sample Size

This part is intended to illustrate the performance of the different tracking schemes by comparing the results of an automated evaluation on the three DARPA Vivid datasets with varying amount of samples. Each scheme is run five times. To compare the different methods, the percentage of successfully tracked frames is computed by dividing the number of successfully tracked frames by the number of total frames in the corresponding sequence. The findings of the evaluation described in Section 6.4.1 are already applied in this experiment. This means, the value of β_3 is set to 0.1 for the CPF scheme and to 0.05 for the other schemes. Figure 6.8 shows the results for sequence *egtest01*. The two horizontal lines indicate the corresponding successfully tracked frames for the KLT tracker (dashed line) and the Mean Shift tracker (dash-dotted line). For each number of samples, the minimum (lower bound), maximum (upper bound) and median (marker symbol) of successfully tracked frames is computed over all corresponding five experimental runs. These values are indicated by the vertical lines. The rising amount of successfully tracked frames with an increasing amount of samples can be clearly observed for all methods. Mean Shift is only able to track 2 % of the frames successfully while the KLT tracker is able to track the full sequence. It is observered that its corner features are limited to the rear of the car which results in a constant deviation from the ground truth. The most tracking failures happen during the initial part of the sequence when the car turns around. A second error source starts at frame 1037, where the camera accelerates and the relative motion of the car increases.

Compared to the other methods, the accuracy of the LBP-TPF tracker differs greatly with its ability to track the full sequence. Although its RMSE values are the lowest when compared with other methods (Table 6.6), it is not able to track all frames of any sequence. Usually, the LBP-TPF tracker loses the target within the first 200-500 frames due to appearance changes of the car the tracker is not able to cope with. As it can be seen in Figure 6.9a, the target is tracked successfully in the first frames. The target and the corresponding likelihood is magnified to the right of the figure clearly shows that the target is still detectable in the frame. As the likelihood value of the object slowly decreases, the target is eventually lost. The likelihood map of the frame depicted in

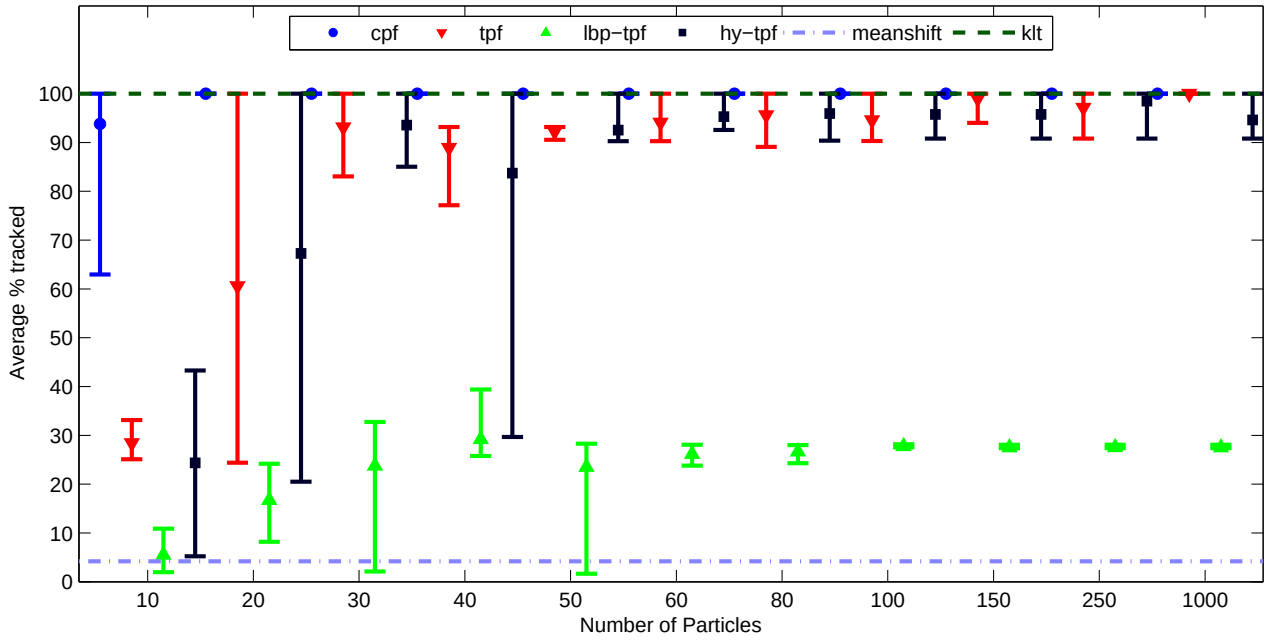
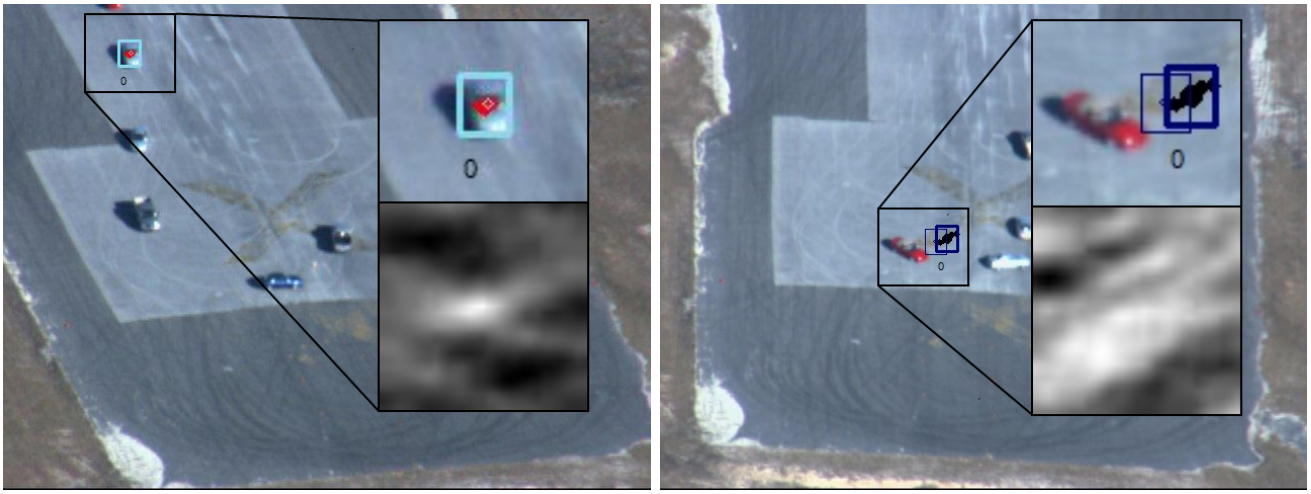


Figure 6.8: Results of automatic evaluation using the *egtest01* sequence.

Sequence (# Frames)	*CPF*	*TPF*	*LBP-TPF*	*Hybrid-TPF*	*Mean Shift*	*KLT*
egtest01 (1810)	2.585	4.355	**1.345***	4.265	3.98*	4.31
egtest02 (1300)	0.91	2.755	**1.865***	2.25	3.98*	3.68
redteam (1917)	3.31	2.605	**1.355***	2.565	2.05*	3.47*
Average	2.27	3.24	**1.52***	3.03	3.37*	3.82*

Table 6.6: Results for the evaluated schemes applied on the object sequences.



(a) (b)

Figure 6.9: Evaluation using LBP-TPF tracker: (a) Initial frames of the Sequence (b) Target is Lost at Frame 580. Images best seen in color.

Figure 6.9b shows that the unique location of the object has vanished and the surrounding background cannot be distinguished from the object anymore.

The CPF tracker it not able to cope with the two obstacles of the *egtest02* sequence and gets distracted by a subsequent car of the first convoy. In this case the target is declared lost. If the amount of samples is increased, the tracker is able to reach the second obstacle of the sequence, which concerns an opposite convoy passing by the convoy of the tracked car. The ability of the template-based trackers to track the sequence increases with the amount of samples used. The main cause of the Hybrid-TPF tracker to fail is the dissociation of the two likelihoods. Both the TPF and Hybrid-TPF tracking scheme are able to track the sequence with 20 samples. Although the influence of the LBP likelihood is only 0.2, it is still able to confuse the tracker. Figure 6.4a depicts such as case where the two maxima of each likelihood are different.

The results of the *redteam* sequence are summarized in Figure 6.10. Considering only results where the method is able to track the full sequence, Hybrid-TPF and TPF provide the best results for 250 (Table 6.6) and 1000 samples. On the other hand, CPF is able to track the full sequence using only 20 samples, with a RMSE value of 4.22. During evaluation, it turned out that all template-based trackers (LBP-TPF, TPF and Hybrid-TPF) have two sections of increased error. After investigation, it turned out that these errors are related to the varying size of the target's bounding box. Figure 6.11 compares the deviation from the initial area of the target with corresponding error of the template-based trackers. As it can be seen, the higher the deviation in any direction (box becoming smaller or greater) the error increases slowly but steadily. This can be explained by the missing width and height variable in the state of the particle filter. Although the template-based trackers do not estimate the bounding box of the object, its RMSE is lower compared to the histogram-based CPF tracker. As soon as LBP features are introduced into the likelihood, the performance degrades as only 1.2 to 15 % of the frames are tracked successfully. The LBP-TPF tracker fails due to the same reason as the other template-based trackers, however, due to the lower likelihood, it does not manage to recover after the box changes back to the size of the target template. Table 6.6 summarizes the RMSE of each tracking scheme when applied to the presented sequences. The values show the median of the average RMSE for 5 runs. The amount of samples for the particle filter are chosen to be 250 because according to the results, smaller values cannot deal with the variance of object appearance and will lead to tracking failures especially for the TPF tracker and the Hybrid-TPF tracker. The

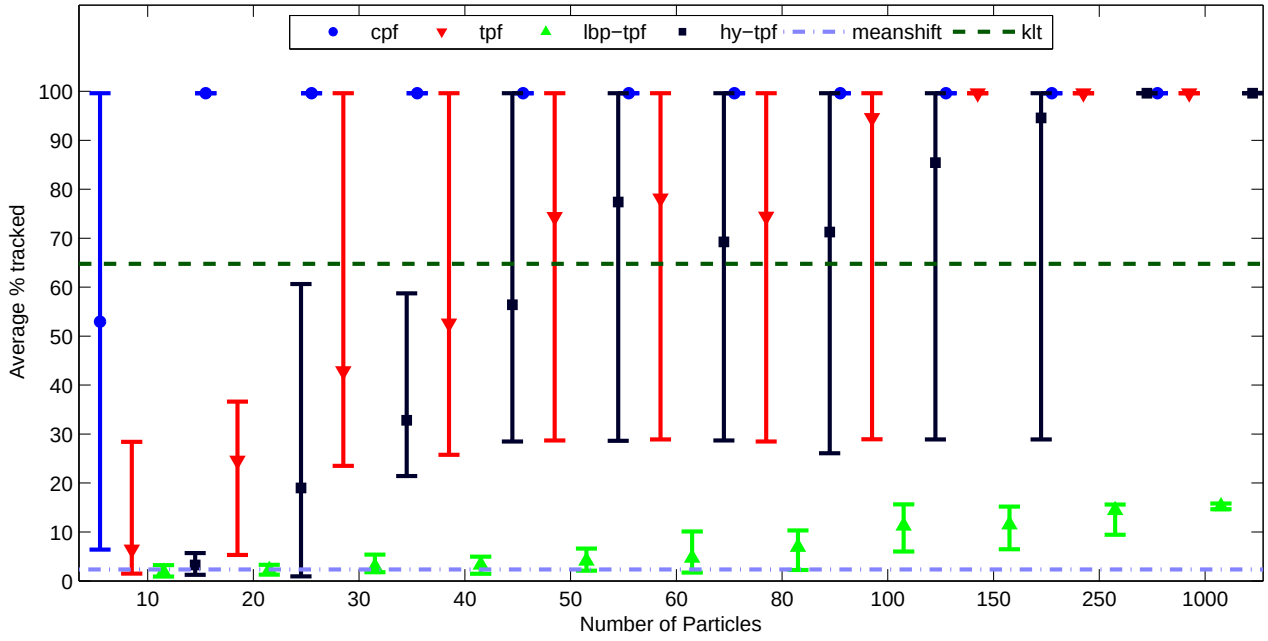


Figure 6.10: Results of automatic evaluation using the *redteam* sequence.

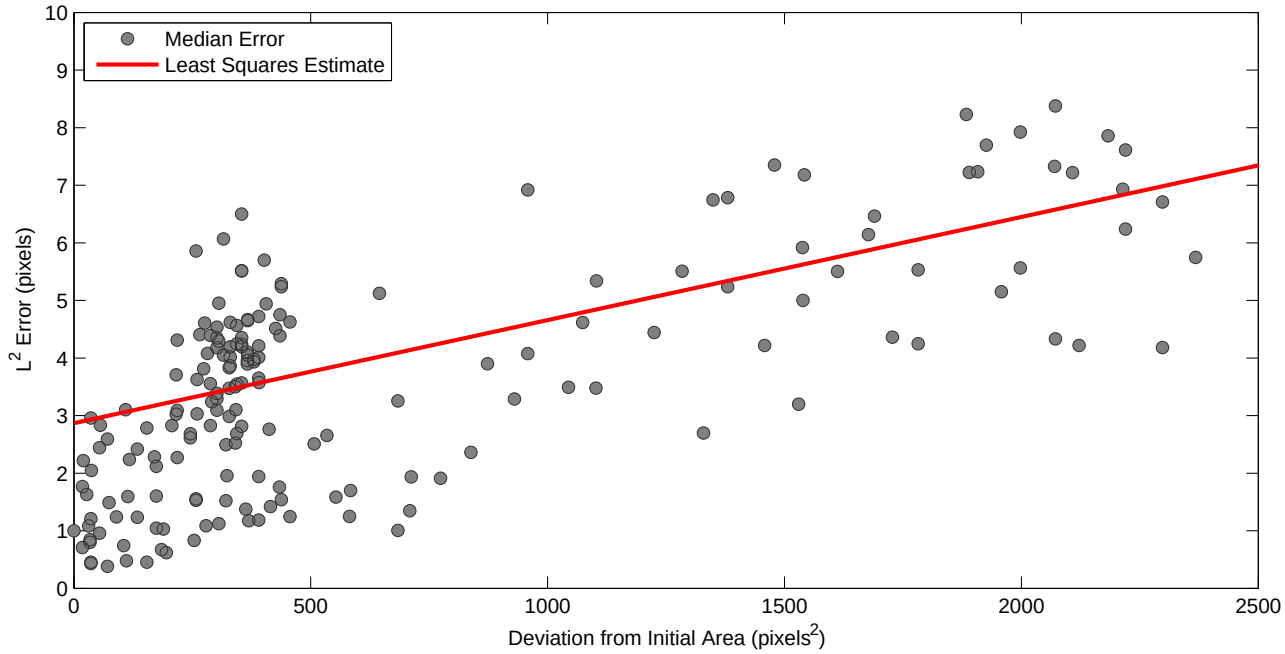


Figure 6.11: Deviation of target box from initial size in comparison with the L^2 distance error.

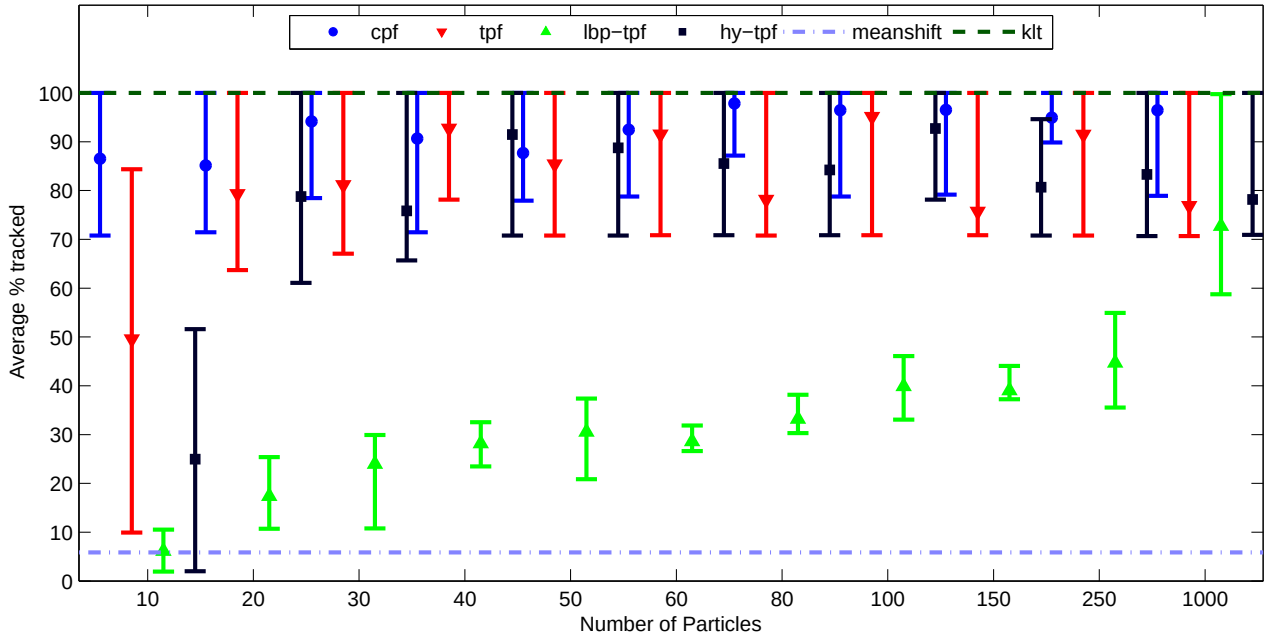


Figure 6.12: Results of automatic evaluation using the *egtest02* sequence.

results marked with star (*) are computed from incomplete sequence results because the method is not able to track the sequence successfully even once. Despite the fact that the LBP-TPF tracker is not able to deal with noise and appearance changes, it provides the most accurate results before it loses the target. Mean Shift loses the target in every sequence after the initial 38-72 frames and is not able to cope with the varying appearance of the objects. The KLT tracker is able to track both the *egtest01* and *egtest02* sequence, but loses the target in the *redteam* sequence after 1200 frames. For all template-based trackers, the main sources of error concern appearance changes and rapidly increasing motion of the object, as expected. Because the target model is only updated when the Gaussian likelihood is within $\sigma \times 0.2$, drastic changes of the candidate model due to noise and clutter effectively prevent any further updates. Interestingly, nearly all methods have the same minimum percentage concerning the different runs as it can be seen in Figure 6.12 and Figure 6.10. The minimum of the failures is in the *egtest02* sequence is caused by frame 919 because the camera is moving to the right very fast. The runs losing the target between 411 and 554 in the *redteam* sequence are caused by the camera zooming out and at the same time rotate to the left from the car's perspective. Additionally, there is a wooden fence crossing the cars appearance model at frame 455, causing any template-based tracker with too little amount of samples to fail. What can be observed during evaluation is that an increase of samples does not necessarily improve the RMSE value towards the target, but merely increases the robustness against occlusion, clutter and rapid motion changes. Interestingly, in every sequence evaluation, there is one run where CPF is able to track the sequence with only 10 samples. This means that although the RMSE might be higher for the *redteam* sequence, the target is represented better by a histogram.

6.4.4 Summary

The automatic evaluation shows some interesting results towards the behavior of the different methods. The motion-based likelihood is necessary to cope with similar targets, such as in the *egtest02* sequence. Nevertheless,

the CPF tracking scheme cannot be improved by this strategy. The system variance of any template-based tracking scheme is lower when run multiple times compared with CPF. The Mean Shift tracker is not able to completely track any sequence. The KLT tracker is able to track the *egtest01* and *egtest02* sequence completely. Interestingly, the performance of LBP-TPF is the best concerning the RMSE, however, only a fraction of frames is tracked. This means that it provides the most accurate estimate of target's location, but it is not robust enough to follow the target in the event of clutter. Because the template-based tracking scheme has no state information concerning the objects size, the error is increased with any changes of the object's scale. Increasing the amount of samples does not increase RMSE but prevents target from being lost and decreases the variance over several runs. To conclude the accuracy results, CPF is able to track most runs without losing the target, except *egtest02* where Hybrid-TPF has less failures. In general, the main source of error concern rapid changes in appearance and fast motion.

6.5 Interactive Facial Tracking

In addition to the automatic evaluation using non-facial objects, the presented methods were applied to five video sequences of different subjects (*S1* to *S5*) with attached landmarks of different color. Depending on the camera setup, a different marker size m is selected manually to ensure that the right region will be tracked. To prevent the various values of Δ and standard deviations of the motion model to be determined manually for each dataset, it has been derived from the size of the markers. Although the marker size does not necessarily represent the pixel motion of the face, it still is a good indicator how large the expected motion will be. This means that every size dependent variable v_s is computed using the following formula:

$$v_s = 2 \times (30/m), \tag{6.4}$$

which is based on an aspect ratio between two sequences of the same real marker size but - due to a different resolution - a different pixel size. All implemented variants have been evaluated using 250 particles to prevent that too little hypothesis are available to estimate the rather complex facial environment. The interactive part is given by the possibility of the operator to interfere with the tracking process as soon as markers are out of place subjectively. Additionally, similar to the automatic evaluation in the previous section, if any of the following conditions hold, the system will require user intervention as well:

- The estimate $\hat{x}_k$ is outside of the segmented facial region (Section 5.1.2). This condition holds true if the color of any estimated marker position $\hat{x}_k$ of has a color intensity value of $[255, 255, 255]$, which is set for the background of the non-facial region during the segmentation step.
- The position of the estimate $\hat{x}_k$ is outside of the image region. This constraint is necessary because the facial region could adjoin the image boundary.

Because during experiments it turned out that the Mean Shift tracker is unable to track the *S5* sequence in the default size, the video size is set twice as large to 1472×1152 using upsampling with subsequent bicubic interpolation. Hence, all subsequent Mean Shift results for this sequence are marked with a star (*) to indicate that a different video size than the original size is used. The names of the markers mentioned in this section correspond with the clinical scheme presented in Section 1.1. A missing value (-) indicates that the corresponding method lost all markers from the first to the second frame, making it impossible to generate any meaningful evaluation result.

6.5.1 Qualitative Comparison

Table 6.7 shows the Average RMSE value of each evaluated sequence. Although the interaction with the operator decreases the RMSE in most cases, it is not possible to

Sequence (# Frames)	*CPF*	*TPF*	*Hybrid-TPF*	*LBP-TPF*	*Mean Shift*	*KLT*
S1 (623)	5.4246	3.9533	**3.8981**	4.5554	6.7502	5.7024
S2 (577)	6.9721	**4.4786**	5.0759	5.6061	7.0118	6.571
S3 (446)	5.618	**4.2517**	4.3161	4.7796	5.2975	5.1718
S4 (176)	4.5246	1.5099	**1.4908**	1.5068	-	-
S5 (1510)	4.7409	3.5981	**3.3437**	4.2916*	13.0623*	5.2695†
Average	5.4560	**3.5583**	3.6249	4.1479*	8.0305*	5.6787†

Table 6.7: Average RMSE of the evaluated methods for each video sequence.

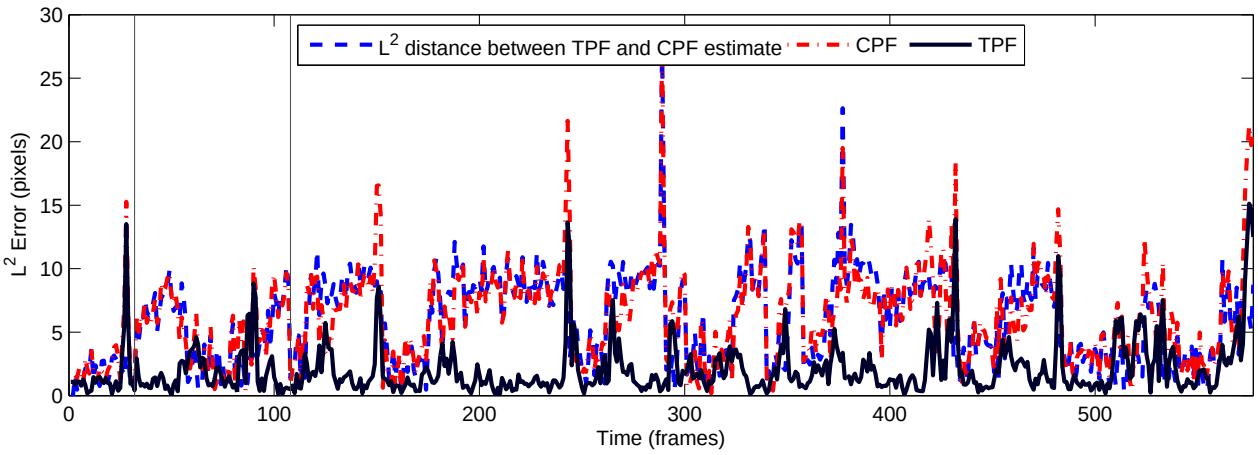


Figure 6.13: Error towards ground truth and the difference between the CPF and TPF tracking scheme.

- place the marker without any deviation from the ground truth and
- remove any other error which is introduced until the failing marker is selected by the operator.

Nevertheless, this value is a good indicator for the accuracy of the method towards the ground truth.

The markers which contribute most to the difference between the RMSE value of the TPF and CPF tracking scheme concern the lower right mouth region. This means that especially in the fast and complex movement of the *S2* sequence, the methods behave differently. Figure 6.13 compares the error between the CPF and the TPF scheme for sequence *S2* and the *RML* marker, which contributes most to the difference in the RMSE between TPF and CPF. As it can be seen, the differences in the RMSE results from the lower distance from the TPF scheme to the ground truth and the distance between CPF and TPF being nearly equal with the CPF schemes's distance to the ground truth. The error curve starts to increase every time when the patient starts to move, e.g. frame 32 (first vertical line) where the first smile expression starts and frame 108 (second vertical line) where the patient moves back to rest. This means that the TPF scheme is more suitable for facial expressions which concern the mouth region, even though the marker will surely deform and change its appearance. This behavior can be also observed in the *S1* scheme. Interestingly the differences for the *S3* sequence mostly concern the eye region, although the patient whistles and the markers *RML* and *LML* are occluded by the lower lip many times.

The baseline trackers KLT and Mean Shift have a slightly better RMSE than the tracking CPF scheme for the *S3* sequence. This can be explained by the fact that Mean Shift uses the same target representation as the

CPF tracker. The KLT tracker has a slightly better RMSE than the CPF scheme for the *S2* sequence. For the *S5* sequence, the accuracy of Mean Shift is the worst, compared with the other methods. The smaller the objects are, the higher is the ambiguity with the background, especially if the ground truth is not correctly placed over the initial marker. Both baseline algorithms have major problems tracking the *S5* sequence, which has different resolution and marker size. The KLT tracker does not generate enough features within the boundaries of the object. Additionally, the strongest features are found at the facial characteristics such as pupils or the mouth corners, which causes them to be tracked instead after some time. The KLT tracker has been evaluated with a marker size of 12 instead of 10 for the *S5* sequence, in order to find enough corner features around the actual position. Additionally, the quality level is set to 0.05 instead of 0.2. This is indicated by the (†) in the corresponding cell in Table 6.7. Despite these settings, it can be observed that corner features drift apart, eventually enlarging the estimated model of the ellipse and introducing error. There are no results for *S4* sequence because neither of the baseline algorithms are able to track it due to the bounding box of the marker being only 3-4 pixels wide. Tracking the sequence with twice the resolution led to the same result. The KLT tracker was not able to track the sequence due to many outliers in the optical flow field, even though there is no perceivable facial motion. Also, motion blur imposes an additional problem for all baseline trackers, resulting in an inability to cope with fast movements. What can be noticed is that all tracking schemes perform better when applied to the *S2* sequence than the *S1* sequence. After brief investigation it turned out that the variance of the x axis is higher than the variance of the y axis of for the *S2* sequence, which opposes the chosen values of σ_y and σ_x. This means that a more sophisticated method is necessary to estimate the movement variables for the particle filter, such as learning the facial motion in advance. Because all particle filtering schemes show a decline of accuracy the more frames are tracked, this issue is investigated in detail by having a look at the error over time for each marker separately. It turned out that the cause of the decline is not a general decline of the error concerning all markers, but specific markers engaging in facial expressions tend to have increasing error towards the ground truth over time, especially these markers where manual intervention is necessary. Effectively, this results in a very large bias on the manual selection of the operator. The LBP tracking scheme is able to immediately reacquire the target when it has been lost in the previous frame. This means that although it is sensitive to changes in appearance it is also able to robustly detect the target in the next frame, when it is still within its search region. The performance of the corresponding tracker is dependent on the affinity of the samples in the particle filter towards the actual target. Hence, the ability of the features to distinguish foreground from background is very important. Figure 6.14 compares a set of three markers where the representation of the target is good with three other markers where the representation is rather poor. Ideally, the likelihood is Gaussian distributed with a unique mode, as it matches with the assumed likelihood distribution defined in Section 5.2 by Equation 5.6. As it can be seen, the three likelihood maps from Figures 6.14a-6.14c, have a high ambiguity whereas the likelihood maps from Figures 6.14d-6.14f provide a unique maximum value with little cluttered background. Although the likelihood which is based on the MRF prevents target hijacking, it can also lead to situations where the target with the higher likelihood pushes away the target of the lower likelihood. These factors need to be investigated and balanced in any future work.

6.5.2 User Interaction

Table 6.8 summarizes the number of user interactions required for each method. The more user interaction are required, the less is the method able to track the sequences automatically without losing the target. To be able to compare the sequences among each other, the second number in the table is the fraction of frames which needed any user intervention.

The differences in the x and y axis variance, as already explored in the previous section, also account in differences concerning the number of interventions. For example, the CPF tracker scheme needs 0 interventions for *S1* sequence, but 20 for the *S2* sequence. The interventions for the *S3* sequence are necessary when the two markers in the lower mouth region get occluded while the patient whistles. In general, two types of interactions can be distinguished, with the first one being necessary if a target is slowly moving away from the actual marker

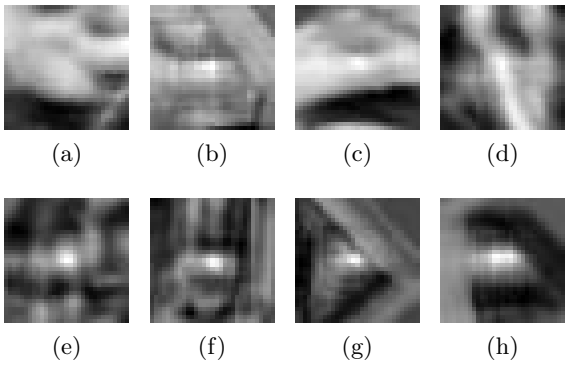


Figure 6.14: Two different types of likelihood images: (a)-(d) targets with ambiguous likelihood; (e)-(h) targets with non-ambiguous likelihood.

Sequence (# Frames)	*CPF*	*TPF*	*Hybrid-TPF*	*LBP-TPF*	*Mean Shift*	*KLT*
S1 (623)	**0**	12/0.02	13/0.02	21/0.03	16/0.02	55/0.09
S2 (577)	20/0.03	**3/0.005**	19/0.03	18/0.03	68/0.12	148/0.26
S3 (446)	**8/0.02**	10/0.02	10/0.02	**8/0.02**	30/0.07	26/0.06
S4 (176)	3/0.02	0	0	0	-	-
S5 (1510)	59/0.04	**32/0.02**	88/0.06	106/0.07	57/0.04*	139/0.09
Average	0.021	**0.013**	0.028	0.03	0.05	0.099

Table 6.8: Number of interventions of the evaluated methods for each sequence. The first value in a cell indicates the absolute number, the second value indicates the number relative to the amount of frames in the sequence.

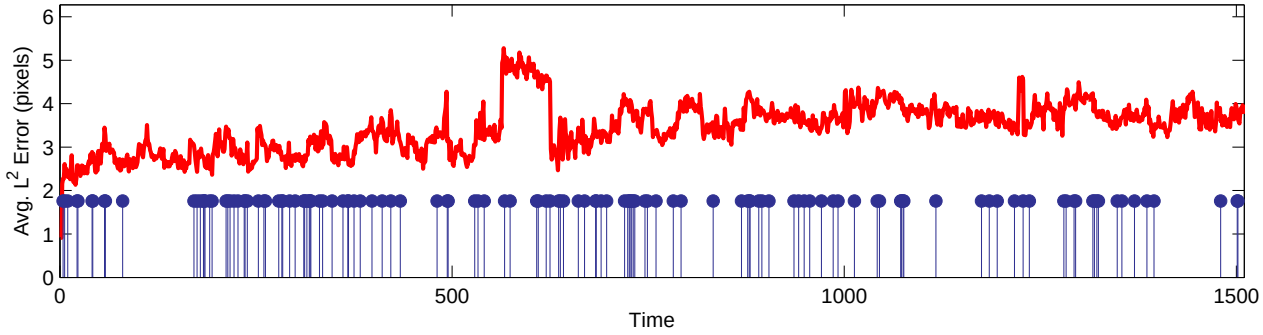


Figure 6.15: Average RMSE compared with user interactions for KLT tracker applied on S5.

and the second one being necessary if the target is completely lost, i.e. it is not covered by the estimate anymore. The operator would have to be highly alert to ensure that the influence of the second type is not disturbing the clinical evaluation process. In the *S1* sequence, CPF has some minor deviations from the target through the sequence, which must be constantly corrected.

Figures 6.15 and 6.16 compare the Average L^2 error with the amount of interventions necessary to correct the tracker. As it can be seen, although the error is very similar, the amount of interventions is much higher for the baseline tracker approach which only relies on optical flow information. This means that compared to the 139 interventions for the KLT tracker, the TPF tracker only required 32 interventions throughout the sequence. Also, while the interventions are equally distributed for the baseline approach, the particle filter based tracker required interventions only in the later parts of the sequence. This can be explained by the fact that although the particle filter is robust over short sequences, the ability of the set of particles to represent the actual positions of the targets degrades over time. Another good example is the sequence *S3*, where the KLT tracker is unable to deal with the occluded markers and needs a permanent intervention for the targets of the lower mouth, while the particle filter scheme only needs a few interventions where the marker is completely invisible. The MRF likelihood (Section 5.2.2), which ensures that interacting targets do not overlap each other, result in negative effects as well. In the *S1* sequence, the patient closes the eyes several times. With every time, the surrounding two pairs of markers approach each other until they eventually fuse. Although the tracked targets of each marker stay separate, the MRF likelihood forces one target to be pushed aside, while the other target tracks the fused pair of markers. This problem accounts for most of the interventions of both the TPF and Hybrid-TPF tracking scheme (Table 6.8). Compared with the template-based trackers, CPF performs better around the eye region of the *S1* sequence. The additional variables of height, width and rotation in the state model is able to be more adaptive towards the changing shape of the markers in this situation, making no interventions necessary over the whole sequence.

The Mean Shift tracking methods requires interventions in cases where the target moves fast, especially when motion blur occurs. This is not an issue with the KLT tracker which uses a pyramidal approach to compute the optical flow maps at different levels of scale. On the other hand, KLT loses its feature points the more frames are tracked, i.e. no targets but single features drift away from the estimated positions. Only the constraint concerning the target area prevents that these features introduce additional error towards the actual location. KLT is not able to track the interacting markers in the *S1* sequence and needs correction. The Hybrid-TPF tracker scheme requires interactions around the eye region of *S1* as well. In average, the least number of interactions for all sequences is required by the TPF method, the most number of interventions are required by the baseline schemes (Table 6.8).

Figure 6.17 shows the distribution of interventions by marker type according to the scheme in Figure 1.2 presented in Chapter 1.

For each marker the total number of interventions is shown along with the contribution of each sequence.

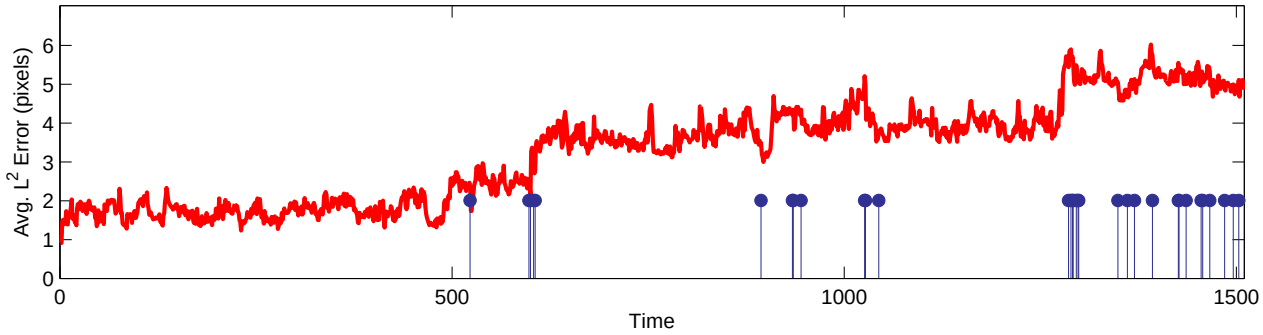


Figure 6.16: Average RMSE compared with user interactions for TPF tracker applied on S5.

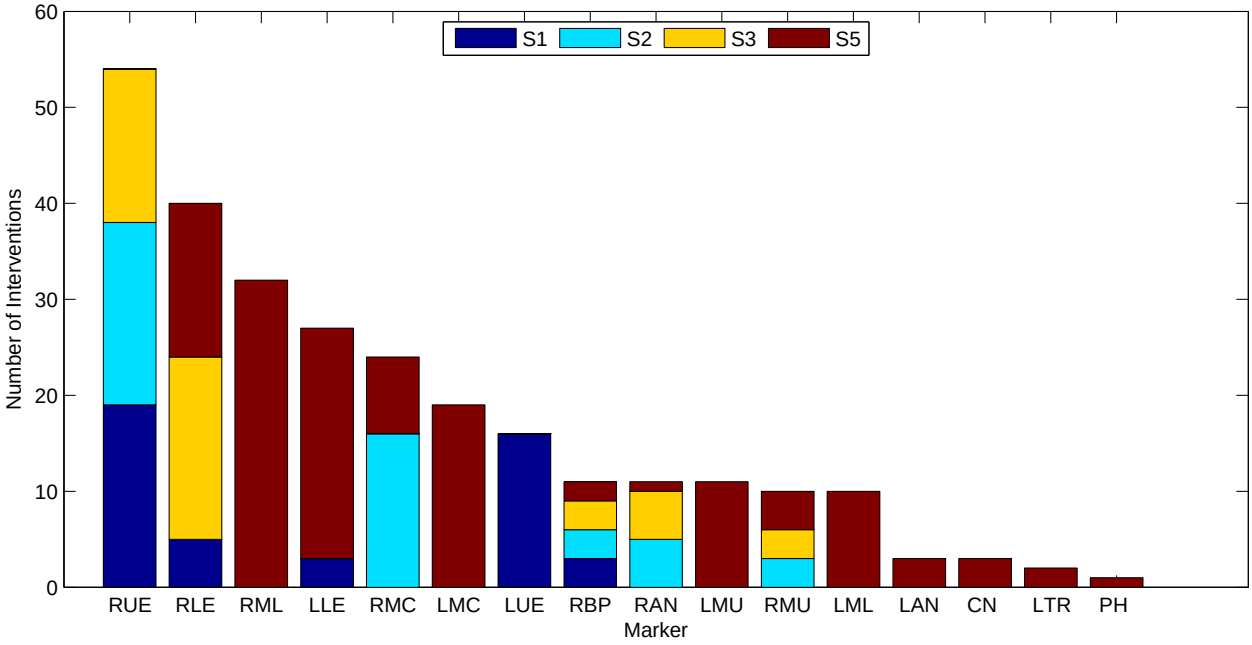


Figure 6.17: Distribution of interventions per marker type.

	CPF	*TPF*	*Hybrid-TPF*	*LBP-TPF*	*Mean Shift*	*KLT*	*Ground truth*
S1 (623)	55	42.21	70.5	65.83	**30.46**	39.05	43
S2 (577)	44.3	39.25	63.17	59.13	**25.36**	46.21	31
S3 (446)	35.6	35.08	52.08	48.05	**19.78**	28.80	41
S4 (176)	1.1	**1.09**	1.62	1.92	-	-	-
S5 (1510)	227.91	161.38	194.32	189.68	**42.13***	40.03	540

Table 6.9: Total tracking times of the evaluated methods for each sequence (minutes).

The markers *RUE* and *RLE* - both at the right eye region - require the most interactions (54 and 40 times, respectively). There are also some markers which are corrected only in the *S5* sequence due to its different camera setup. This concerns the *RML* and the *LLE* markers, which are located on the right lower mouth and the left lower eye region, respectively. Interestingly, the *RMC* marker which concerns the smile expression in sequence *S2* is lost 16 times while the opposite marker *LMC* is never corrected.

6.5.3 Sequence Runtime

In this subsection, the methods are compared by the time they need to track the sequences. Additionally, they are compared with the time to create the corresponding ground truth manually. As ground truth creation essentially boils down to locating all markers in every frame by hand, this is an important factor to demonstrate that the evaluated methods are able to speed up the clinical evaluation. Table 6.9 summarizes the total tracking times of each sequence and tracking method. The computationally most expensive tracking schemes are the CPF and the Hybrid-TPF method. The CPF scheme has to compute a histogram for every sample, which takes up most of the time to track a single frame. The Hybrid-TPF scheme has to compute two likelihoods, one for each cue (intensity and LBP) for each frame. Additionally, the MRF likelihood which needs to compute the overlap among all markers requires additional computational effort because it needs to compute a graph of all targets and the edge costs based on the overlap among the targets. Nevertheless, it is possible to reduce the tracking time from previously 540 minutes or 9 hours to 161 minutes or approximately 2.5 hours, which is a decrease of 70.2 %. Since there is no groundtruth available for a full cinically relevant sequence, only an estimate is given towards the tracking time using the presented methods. Assuming the time to manually track the clinical sequences to be 5 hours, it would result in an estimated tracking time of approximately 89 minutes. Given the fact that the implementations are not optimized and partly computationally inefficient, this value can be even more decreased. A gain of execution speed can be also expected when the particle filter algorithm would be executed in parallel. This would also also allow a higher amount of samples to be chosen per target, which would directly decrease the RMSE of the algorithm as well. Mean Shift needs the least time to track the sequences because only one hypothesis (the target histogram) needs to be evaluated.

6.5.4 Summary

This evaluation outlines the performance of the different tracking schemes (Table 6.5) on data consisting of facial expressions. All particle filter methods prove to be advantageous when compared with the baseline methods, as there are 2 to 3 times less interventions necessary with a comparably equal error towards the actual ground truth. When considering both RMSE and number of interactions, the TPF tracking scheme works best over all evaluated sequences. The accuracy of Mean Shift and the CPF scheme is comparable because they both use a histogram to represent the target. The static assumption of the variance of movement towards the image axis turned out to be a limiting factor in the effective generation of samples during tracking, which is a big issue and needs to be dealt with in the future. The baseline schemes are not able to track any sequence of the *VMU* camera setup, even if the resolution is doubled. Due to the different accuracy and number of user interactions

it is shown that the selection of the features and the target representation are the key issues together with an appropriate model of facial movement. The better the features in terms of differentiation between markers and facial background, the less samples can be used, which will in turn affect the tracking time. Compared with manually tracking the sequences frame by frame, the time is reduced by 70 % with only a fraction of necessary user interactions. Although the employed Voronoi Tesselation in combination with the MRF likelihood prevents targets from being hijacked, possible side effects might compensate this improvement by pushing target estimates away from the actual marker. There as tradeoff between lower interactions and higher accuracy. While the CPF tracking scheme provides lower accuracy compared with the Hybrid-TPF method, it requires a higher attention due to interactions.

Chapter 7

Conclusion

This chapter is intended to both conclude and summarize this thesis (Section 7.1). Additionally, current shortcomings and their possible improvements are discussed (Section 7.2).

7.1 Summary

In this thesis, we have analyzed the possibility of employing a semi-automatic tracking scheme on the problem of tracking facial markers. As the particle filter provides a framework to combine different cues, several tracking schemes have been modeled in order to evaluate their accuracy when applied on the presented scenario. The particle filter uses a set of samples which are constantly weighted based on their likelihood of being drawn from the current image and the associated target models of the markers. A *template-based tracking* (TPF) scheme was formed by using a normalized cross correlation approach to compute this likelihood. A similar scheme is based on *Local Binary Patterns*, which encode textural information of the target instead of merely their appearance. To compare the template-based approach with histograms, a *histogram-based scheme* (CPF) scheme was formed, in which the target model is represented by an histogram of weighted HSV intensities. In addition to these cues, a deterministic component was added to incorporate object motion. This motion cue is based on the estimated optical flow map between the current and the previous frame. Finally, a third aspect tries to cover scenarios where markers could possibly overlap. Using this framework, the location of the markers can be accurately estimated in every frame and the identity of each marker is preserved using a combinatorial data association method. This automatic approach has been extended in such way to allow the operator to intervene at any time in the tracking process to make sure that possible target losses are prevented and the error is minimized.

The main contribution of this thesis forms a comprehensive evaluation of the presented methods, where its results have also been published at the 21st conference of the International Association for Pattern Recognition (ICPR) [36]. At first, the different schemes have been used to investigate the behavior concerning their variance, the importance of the motion-based likelihood and the performance towards non-facial landscape scenes. Additionally, the methods have been compared with the Mean Shift tracker and the KLT Tracker, which only rely on a single hypothesis concerning object's location. This evaluation shows some interesting results towards the behavior of the different methods. The motion-based likelihood turned out to be an important factor to cope with similar targets, such as in the *egtest02* sequence. The variance of any template-based tracking scheme is lower than the scheme using color-based histograms. The Mean Shift tracker is not able to completely track any of the object sequences. The KLT tracker is able to track the *egtest01* and *egtest02* sequence completely. Additionally, it was discovered that the template-based trackers have increased error in parts of the sequence where the object's size deviates from its initial bounding box. Increasing the amount of samples does not increase RMSE but prevents target from being lost and decreases the variance over several runs. In general, the main source of error concern rapid changes in appearance and fast motion. Concerning the Hybrid-TPF tracker

scheme, the combination of the intensity-based likelihood and the LBP-based likelihood often leads to different maxima of the likelihood, which has been observed as the major problem in both evaluation parts. An additional problem was the assumption of the variance of movement towards the two axis of the image, as it turned out that often the distribution of samples in the x and y space does not match the actual movement. Due to the different accuracy and number of interventions it is shown that the selection of the features and the target representation are the key issues together with an appropriate model of facial movement. The better the features are in terms of differentiation between markers and facial background, the less samples can be used, which will in turn affect the tracking time. The employed Voronoi Tesselation in combination with the MRF likelihood prevents targets from being hijacked. While it was initially assumed that the particle filter is robust enough to handle occlusions in the mouth region, this was not the case. Although the TPF tracker was able to handle them best, it still required interactions by the user. The CPF scheme has a low rate of interventions, but its accuracy is only comparable with Mean Shift, because in both cases, a histogram is used to represent the target, which lacks any spatial distribution. In both evaluation parts, the performance of the LBP-based tracking scheme provides high accuracy on the cost of a high number of interventions. From the results it can also be concluded that the additional variables concerning size and angle of the target, did only result in improvements where the variance of size is relatively high, such as in the non-facial sequences. On the other hand, this extended model did not result in any improvements concerning the facial sequences.

To conclude this summary, this thesis contributes by examining the performance of these tracking schemes when applied on this clinical scenario. To improve the accuracy and reset lost markers, the clinical operator can interact with the tracking system. It was shown that the chosen methods are superior concerning both number of interactions and error when compared with trackers which employ only a single hypothesis concerning the marker locations. Additionally, it was shown that the evaluated scheme is able to replace the task of manual tracking while preserving a high accuracy. As a result, the time to locate the markers was decreased by 70.2 % with an accuracy of 3-4 pixels towards the available ground truth. When considering all tracked frames in the second part of the evaluation, only 1.3 % of the evaluated frames required an intervention by the operator.

7.2 Future Work

During the extensive period of this thesis, many ideas and extensions came up, some of them small, some of them representing a completely different approach. Future investigation might start with different representations of the Bayesian state space. As the incorporation of the marker width, height and angle did not result in any major changes, the first step would be to investigate if this lack of improvement is caused by the target representation. Additionally, an improved joint tracking scheme is necessary to model any interactions between markers efficiently. Another possible improvement concerns the online learning of facial movement, as currently the assumed motion model does not accurately reflect the typical facial movement. The MCMC (Markov-Chain Monte Carlo) approach from Section 4.4.3 is one of these tracking schemes. Because the approach does not rely on a clustered state-space to track multiple objects simultaneously, but models a joint state-space instead, different interaction constraints could be incorporated easily. In a preliminary experiment (Figure 5.2), it seemed possible to track two pairs of interacting markers using the Metropolis-Hastings algorithm (Algorithm 4.1). Nevertheless, if as many as 15 or 34 markers should be tracked, even a multiple of the 2000 samples have not been enough to prevent target loss, which leads to one of the first possibilities to improve this work.

Concerning the chosen features, other approaches used saliency maps [56] and SIFT features [38] could be considered as well. Despite these rather small improvements which concern the existing 2D approach, the most important suggestion for any future work is to move to 3D information, which has been already prepared for the given scenario in the bachelor thesis by M. Cerman *'Camera Calibration Methods using a Multi-Mirror Setup for a Medical Environment'* , Vienna University of Technology, 2012. The correspondences between the mirror images can be used to track the markers in 3D and could improve the accuracy towards 2D tracking by 57 % [21].

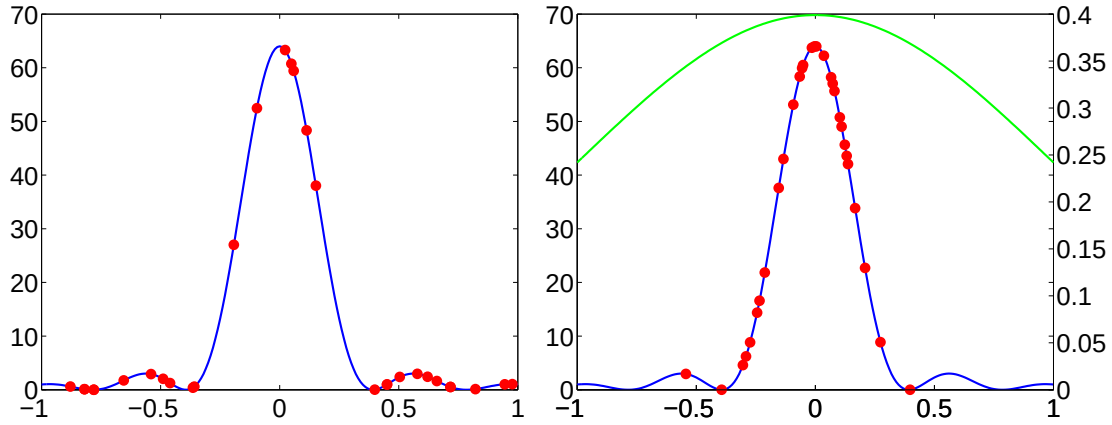
Appendix A

Sampling Example

This appendix is intended to give a short example on the sampling methods introduced in Section 4.4. *Uniform sampling* $U(a, b)$ samples from the distribution $P(x) = 1/V$, where V is the volume of the distribution within interval $[a, b]$. In this case, it is both possible to sample from the distribution as well as to evaluate it. Uniform sampling can also be seen as a random number generator within interval $[a, b]$.

$$f(x) = \begin{cases} 64 & : x = 0 \\ (\sin(8x)^2)(xs^{-2}) & : x \neq 0 \end{cases} \tag{A.1}$$

Figure A.1 compares uniform sampling with importance sampling using Equation A.1. The real integral value of the depicted function is 24.0. In each case, 30 samples are drawn using uniform sampling and importance sampling. For importance sampling each sample x_i is distributed according to the importance distribution $x_i \sim N(0, 0.25^2)$, The resulting estimated area is then 14.7 for uniform sampling compared to 25.8 for importance sampling. This can also be seen when comparing Figure A.1a with Figure A.1b. According to the importance weight $Q(x)$, which is shown green in Figure A.1a, samples having a higher probability are more likely to be chosen. Obviously, the choice of the importance distribution $Q(x)$ is crucial. In both cases, the more samples will be used, the higher will be the accuracy of the estimate area. Basically, uniform sampling is importance sampling with $Q(x) = 1/V$ [50].



(a) Uniform Sampling (b) Importance Sampling

Figure A.1: A comparison between uniform sampling (a) and importance sampling (b) with $Q(x) \sim N(0, 0.25^2)$ applied to Function A.1.

Bibliography

[1] K. R. T. Aires, A. M. Santana, and A. A. D. Medeiros. Optical flow using color information: preliminary results. In *Proceedings of the ACM Symposium on Applied Computing*, SAC '08, pages 1607–1611, New York, NY, USA, 2008. ACM.

[2] F. Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Survey*, 23(3):345–405, 9 1991.

[3] Y. Bar-Shalom and X. R. Li. *Estimation and Tracking: Principles, Techniques, and Software.* Artech House, Boston, MA, 1993.

[4] J. Barker. Tracking facial markers with an adaptive marker collocation model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 665–668, 18-23, 2005.

[5] S. M. Bhandarkar and X. Luo. Integrated detection and tracking of multiple faces using particle filtering and optical flow-based elastic matching. *Computer Vision and Image Understanding*, 113(6):708–725, 2009.

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

[7] J. Bouguet. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm. *Intel Corporation Microprocessor Research Labs*, 2000.

[8] G. E. Box and M. E. Muller. A Note on the Generation of Random Normal Deviates. *The Annals of Mathematical Statistics*, 29(2):610–611, 1958.

[9] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 3 2011.

[10] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnorr. Variational optical flow computation in real time. *IEEE Transactions on Image Processing*, 14(5):608–615, may 2005.

[11] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[12] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of IEEE Conference on on Computer Vision and Pattern Recognition*, pages 142–149, 2000.

[13] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 5 2003.

[14] S.E. Coulson, G.R. Croxson, R.D. Adams, and O'Dwyer N.J. Reliability of the 'sydney,' 'sunnybrook,' and 'house brackmann' facial grading systems to assess voluntary movement and synkinesis after facial nerve paralysis. *Otolaryngology - Head and Neck Surgery*, 132(4):543– 549, 2005.

[15] J. R. Delannoy and T. E. Ward. A preliminary investigation into the use of machine vision techniques for automating facial paralysis rehabilitation therapy. In *Proceedings of Signals and Systems Conference*, pages 228–232, june 2010.

[16] J. Finsterer. Management of peripheral facial nerve palsy. *European Archives of Oto-Rhino-Laryngology*, 265:743–752, 2008. 10.1007/s00405-008-0646-4.

[17] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[18] A. W. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least squares fitting of ellipses. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 1, pages 253–257. IEEE Comput. Soc. Press, August 1996.

[19] M. H. Ghaeminia, A. H. Shabani, and S. B. Shokouhi. Adaptive motion model for human tracking using particle filter. In *Proceedings of the 2010 20th International Conference on Pattern Recognition*, pages 2073–2076, 2010.

[20] P. Giovanoli, C.-H. J. Tzou, M. Ploner, and M. Frey. Three-dimensional video-analysis of facial movements in healthy volunteers. *British Journal of Plastic Surgery*, 56(7):644–652, 2003.

[21] Moffatt K. S. Gross M. M., Trotman C. A. A comparison of three-dimensional and two-dimensional analyses of facial motion. *Angle Orthod*, 66(3):189–94, 1996.

[22] Chris Harris and Mike Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[23] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[24] S. He, J.J. Soraghan, B.F. O'Reilly, and D. Xing. Quantitative analysis of facial paralysis using local binary patterns in biomedical videos. *IEEE Transactions on Biomedical Engineering*, 56(7):1864–1870, 7 2009.

[25] B. K. P. Horn and Brian G. S. Determining optical flow. *Artifical Intelligence*, 17:185–203, 1981.

[26] J. W. House and D. E. Brackmann. Facial nerve grading system. *Otolaryngol Head Neck Surgery*, 93:146–147, 1985.

[27] M. Isono, K. Murata, H. Tanaka, M. Kawamoto, and H. Azuma. An objective evaluation method for facial mimic motion. *Otolaryngology - Head and Neck Surgery*, 114(1):27–31, 1996.

[28] P. C. Johnson, H. Brown, and Jr. Kuzon, W. M. Simultaneous quantitation of facial movements: The maximal static response assay of facial nerve function. *Annual Plastic Surgery*, 32-2:171–180, 1994.

[29] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME-Journal of Basic Engineering*, 82(82):35–45, 1960.

[30] Z. Khan, T. Balch, and F. Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *Proceedings of the 8th European Conference on Computer Vision*, pages 279–290, 2004.

[31] H. King and D. A. Forsyth. How does condensation behave with a finite number of samples? In *Proceedings of the 6th European Conference on Computer Vision*, pages 695–709, 2000.

[32] Y. Koji, I. Inokuchi, M. Maeta, S. Kawakami, and Y. Masuda. Evaluation of facial palsy by moiré topography index. *Otolaryngology - Head and Neck Surgery*, 117(5):567–572, 1997.

[33] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[34] S. Z. Li. Markov random field models in computer vision, 1994.

[35] J.-J. J. Lien, T. Kanade, J. Cohn, and C. Li. Detection, tracking, and classification of subtle changes in facial expression. *Journal of Robotics and Autonomous Systems*, 31:131–146, 2000.

[36] P. Limbeck, W. G. Kropatsch, and Y. Haxhimusa. Semi-automatic tracking of markers in facial palsy. In *Proceedings of 21st International Conference on Pattern Recognition (to appear)*, 2012.

[37] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: Dense correspondence across different scenes. In *Proceedings of the 10th European Conference on Computer Vision*, pages 28–42, Berlin, Heidelberg, 2008. Springer-Verlag.

[38] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 11 2004.

[39] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[40] E. Maggio and A. Cavallaro. Hybrid particle filter and mean shift tracker with adaptive transition model. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 221–224, 2005.

[41] E. Maggio and A. Cavallaro. *Video tracking: theory and practice*. Wiley, 2011.

[42] N. Martins and J. Dias. Visual inspection based on mirror images. In *Proceedings of the 7th International Confernece on Systems on Intelligent Robotic Systems*, 1999.

[43] K. Nummiaro, E. Koller-Meier, and L. Van Gool. An adaptive color-based particle filter, 2002.

[44] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[45] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, jul 2002.

[46] M. Park, J. M. Seo, and K. S. Park. Pc-based asymmetry analyzer for facial palsy study in uncontrolled environment: A preliminary study. *Computer Methods and Programs in Biomedicine*, 99(1):57–65, 2010.

[47] H. Popat, S. Richmond, L. B. Benedikt, D. Marshall, and P. L. Rosin. Quantitative analysis of facial movement-a review of three-dimensional imaging techniques. *Computerized Medical Imaging and Graphics*, 33(5):377–383, 2009.

[48] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2 edition, 1992.

[49] K. M. Rösler, M.R. Magistris, F. X. Glocker, A. Kohler, G. Deuschl, and C.W. Hess. Electrophysiological characteristics of lesions in facial palsies of different etiologies. a study using electrical and magnetic stimulation techniques. In *Electroencephalogr Clin Neurophysiology*, 1995.

[50] R. Y. Rubinstein and D. P. Kroese, editors. *Simulation and the Monte Carlo Method (Wiley Series in Probability and Statistics)*. John Wiley & Sons, 2 edition, 2007.

[51] M. Sabet, R. A. Zoroofi, K. S. Niiat, and M. Sabbaghian. Automated face tracking with self correction capability. In *Proceedings of International Conference of Soft Computing and Pattern Recognition*, pages 280–284, oct. 2011.

[52] J. Sakaeda, T. Tanaka, and T. Kunihiro. Evaluation method of facial palsy using features from 3 dimensional construction. In *International Joint Conference SICE-ICASE*, pages 3692–3695, 8 2006.

[53] M. Sanjeev Arulampalam, S. Maskell, and N. Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[54] J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:593–600, 1994.

[55] M. Shreve, N. Jain, D. Goldgof, S. Sarkar, W.G. Kropatsch, C.-H. J. Tzou, and M. Frey. Evaluation of facial reconstructive surgery on patients with facial palsy using optical strain. In *Proceedings of the 14th International Conference on Computer Analysis of Images and Patterns - Volume Part I*, CAIP'11, pages 512–519, Berlin, Heidelberg, 2011. Springer-Verlag.

[56] D. Sidibé, D. Fofi, and F. Mériaudeau. Using visual saliency for object tracking with particle filters. In *Proceedings of the European Signal Processing Conference*, pages 1776–1780, Aalborg, Denmark, 2010.

[57] J. Sullivan and J. Rittscher. Guiding random particles by deterministic search. In *Proccedings of the 8th International Conference on Computer Vision*, pages 323–330, 2001.

[58] T. Tanaka, S. Orukawa, N. Yoshida, and T. Kunihiro. Quantitative evaluation of facial palsy by contour extraction of facial features. In *Proceedings of he 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society.*, volume 4, pages 3735–3738, 2001.

[59] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the 6th International Conference on Computer Vision*, pages 839–846, jan 1998.

[60] S. Tsekeridou and I. Pitas. Facial feature extraction in frontal views using biometric analogies. In *Procedings of the 9th European Signal Processing Conference*, pages 315–318, 1998.

[61] C. H. Tzou, I. Pona, E. Placheta, A. Hold, M. Michaelidou, N. Artner, W. Kropatsch, H. Gerber, and M. Frey. Evolution of the 3-dimensional video system for facial motion analysis: Ten years' experiences and recent developments. *Annual Plastic Surgery*, 7 2011.

[62] G. S. Wachtman, J. F. Cohn, and J. M. VanSwearingen. Automated tracking of facial features in patients with facial neuromuscular dysfunction. *Plastic and Reconstructive Surgery*, 109(5):1124–1133, 2001.

[63] M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models (Springer Series in Statistics)*. Springer, 2nd edition, February 1997.

[64] L. Wilson-Pauwels. *Cranial Nerves: Anatomy and Clinical Features*. Mosby, 1988.

[65] Y.-T. Wu, T. Kanade, and C.-C. Li. Optical flow estimation using wavelet motion model, 1998.

[66] J. Xie, S. Khan, and M. Shah. Automatic tracking of escherichia coli bacteria.

[67] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Survey*, 38(4):13, 2006.

[68] H. Zhou, Y. Yuan, and C. Shi. Object tracking using sift features and mean shift. *Computer Vision and Image Understanding*, 113(3):345–352, 2009. Special Issue on Video Analysis.

[69] Z. Zhu and Q. Ji. Robust pose invariant facial feature detection and tracking in real-time. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*, ICPR '06, pages 1092–1095, Washington, DC, USA, 2006. IEEE Computer Society.