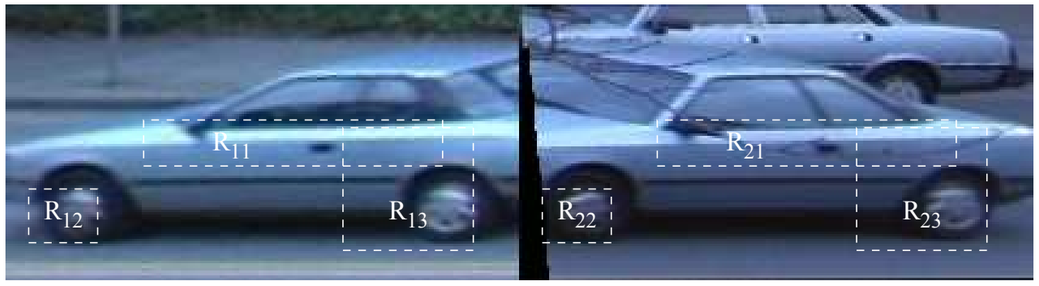
Reidentification of ground targets in EO/IR-imagery

JÖRGEN KARLHOLM, JOAKIM RYDELL, KARL-GÖRAN STENBORG,
MORGAN ULVKLO, JONAS NYGÅRDS, FREDRIK HEMSTRÖM



FOI
Defence Research Agency
Information Systems
P.O. Box 1165
SE-581 11 Linköping

Phone: +46 13 37 80 00
Fax: +46 13 37 81 00

www.foi.se

FOI-R--2586--SE Technical report
ISSN 1650-1942 October 2008

Information Systems

Jörgen Karlholm, Joakim Rydell,
Karl-Göran Stenborg, Morgan Ulvklo,
Jonas Nygårds, Fredrik Hemström

Reidentification of ground targets in EO/IR-imagery

Titel	Återigenkänning av markmål i elektrooptiska sensordata
Title	Reidentification of ground targets in EO/IR-imagery

Rapportnr/Report no	FOI-R--2586--SE
Rapporttyp Report Type	Teknisk rapport Technical report
Sidor/Pages	44 p
Månad/Month	Oktober/October
Utgivningsår/Year	2008
ISSN	ISSN 1650-1942
Kund/Customer	
Forskningsområde Programme area	4. Sensorer och signaturanpassning
Delområde Subcategory	42 Sensorer 42 Above surface Surveillance, Target acquisition and Reconnaissance
Projektnr/Project no	
Godkänd av/Approved by	

FOI, Totalförsvarets Forskningsinstitut	FOI, Swedish Defence Research Agency
Avdelningen för Informationssystem	Division of Information Systems
Box 1165	P.O. Box 1165
581 11 Linköping	SE-581 11 Linköping Sweden

Sammanfattning

Rapporten beskriver aktuell status på utvecklingen av ett ramverk för återigenkänning av markmål i spaningsdata från flygburna sensorer som utvecklas vid FOI. Arbetet utförs inom projekten "ARCUS" och "Autonom Spaning". En komplex signalbehandlingskedja diskuteras i rapporten. Denna kedja innehåller följande delmoment: detektion, multimålföljning, visuell målföljare, segmentering, vyregistrering, egenskapsextraktion, återigenkänning, objektdatabas, planerare och operatörsinteraktion.

Rapporten beskriver statusen för följande fyra arbeten:

- Utveckling av en initial återigenkänningsansats som baseras på CCM (color co-occurance matrices). Denna ansats har verifierats mot simulerade sensordata inom det visuella våglängdsbandet.
- Integration av denna initiala ansats mot multimålföljning via HLA-gränssnitt inom MSS-lab vid FOI.
- Utveckling av en ny ansats som baseras på inlärning av effektiva sammanvägningar av deskriptorer för form och färg. Denna ansats har verifierats mot en internationell databas med tusentals högupplösta fotografier inom det visuella våglängdsbandet av hundratals civila bilmodeller. Klassificeringsprestanda på denna nya ansats bedöms vara världsledande.
- Ett fältförsök har genomförts vid FOI i Linköping för att systematiskt registrera ankommande bilar på parkeringsplatsen vidFOI. Fordonen har registrarats med ett flertal optiska sensorer inom de visuella och termiska våglängdsbanden. Fältförsöket beskrivs övergripande i rapporten och en ansats till annotering av sensordata ges.

Nyckelord: Spaning, övervakning, UAV, återigenkänning, detektion, målföljning, bildanalys

Summary

This report describes the current status of a framework for reidentification of ground targets in aerial surveillance data being developed at FOI. The work has been conducted within the project "ARCUS" and "Autonomous Surveillance". A complex signal processing chain containing the following subparts is discussed: detection, multi-target tracking, visual tracker, segmentation, view alignment, feature extraction, reidentification, object database, planner, and operator interaction.

The report describes the status of four areas:

- The implementation of an initial reidentification method based on CCM (color co-occurance matrices).This work has been verified against simulated sensor data in the visual range.
- Integration of the initial reidentification method with modules for multi-target tracking using HLA.
- The development of a novel reidentification method based on learning effective combinations of local descriptors of shape and color. This method has been evaluated against an international database containing thousands of color images of hundreds of civilian cars. Based on test results, we believe the performance of this method to represent the state-of-the-art in reidentification.
- A field trial has been conducted to systematically register cars entering the parking lot at FOI. Each car has been registred from different orientations using several optical sensors within the thermal and visual ranges. The field trial and an annotation method for collected data are described.

Keywords: Surveillance, reconnaissance, reidentification, UAV, detection, tracking, image analysis

Contents

1 Introduction

1.1 Background

The need for autonomous on-board sensor data processing and sensor management will increase in future aerial surveillance and reconnaissance systems. This arises from the constantly growing quantity of sensors and associated raw data, as well as limitations in communication bandwidth and processing capacity of human sensor operators. Imaging sensors are widely used in surveillance and reconnaissance systems, and increasingly in guided weapons and warning systems. Several basic functionalities of autonomous aerial surveillance systems, e.g. target geolocation, robust navigation, collision avoidance, route and viewpoint planning all require advanced visual capabilities such as target and landmark detection and recognition, scene topography estimation, and image-motion computation.

A signal processing framework for autonomous UAV surveillance has been developed at FOI during the last years. Our working hypothesis is that integration of the detection-tracking-recognition chain with spatial awareness makes possible intelligent autonomous data acquisition by means of active sensor control and path planning. One central research question in the development of such framework is how to be able to *reidentify* earlier detected ground targets.

1.2 Reidentification

In this report we are concerned with methods answering the question "Is this the same object (instance) that was previously observed?". While this may be regarded as a special case of object identification, we prefer to use the term *reidentification*, which is well-established in the field of traffic surveillance. Objects to detect and reidentify are typical ground vehicles, such as military vehicles and cars.

Reidentification differs from general object recognition (categorization and identification) in that, typically, in the former only a single previous (reference) observation is available of the object of interest, whereas in the latter it is often assumed that a comprehensive signature database can be assembled, covering all possible appearances of the object. Since, *e.g.*, object pose and illumination conditions may be significantly different when the object is observed again, matching methods used in reidentification must be highly robust to such variations.

Reidentification is a general problem and can be used to identify people, vehicles or other objects of interest. The scope of this report, however, is limited to reidentification of vehicles using airborne sensors.

1.3 Two methods

Two reidentification methods are described in this report. The first method is aimed at quickly closing the detection-tracking loop shown in figure 1.1. This well-established method, based on colour histogram matching, is described in chapter 4. A second (new) method, described in chapter 5, uses statistical learning from training data to find discriminative image features which are matched using an optimal metric. The

latter method is not yet integrated into the framework given in figure 1.1, but has very promising performance.

1.3.1 Time aspects and sensor data variation

The typical time span between the first detection of a target, until the reidentification step, depends on the surveillance scenario. In this work we assume that a typical time span is from seconds into a couple of minutes. The selected object representation has to handle natural variation in sensor data, such as:

- Variations in relative orientation and scale between the sensor on the surveillance platform and the object to be reidentified. This can be at least partly supported by prior knowledge from the platform navigation system and terrain or road data from a Geographical Information System (GIS).
- Colour variations depending on natural variation in lightning condition (visual sensor).
- Variations of the thermal radiance of the vehicle due to changes in thermal condition of the object, i.e. heated targets tracks, friction in wheels, heated engine and exhausts.
- Variations in specular reflection.
- Partial occlusion depending on scene interaction between the object and the local neighborhood.
- Interaction with the surrounding neighborhood in the image, due to non-perfect object/background segmentation (figure/ground segmentation) caused by e.g. background clutter.
- Different shadow conditions, e.g. changes in cloudiness or changes in relative orientation to the sun.

1.4 Reidentification processing scheme

Figure 1.1 shows the basic components of the reidentification system. A more detailed descriptions of each component is given in chapter 2.

Detector The detection step points out positions in the imagery with a high degree of confidence to contain a moving or stationary ground target (typical a vehicle). Our inhouse developed method for detection of stationary ground targets is presented in [12].

Multi-target tracker The multi-target tracker handles uncertainties of targets tracks and establish, from at statistical point of view, correspondences between detections and target tracks. The reidentification step is tightly interleaved with the association step in the multi-target tracker.

Visual tracker The visual tracker is capable of tracking a detected target in a video or infrared image stream (sequence). The implemented method can handle some appearance variations of the target signature, such as affine deformation (scaling, rotation and skewing) and partial occlusion. This step is also responsible for the object/background segmentation of moving targets. Details are given in chapter 3.2.

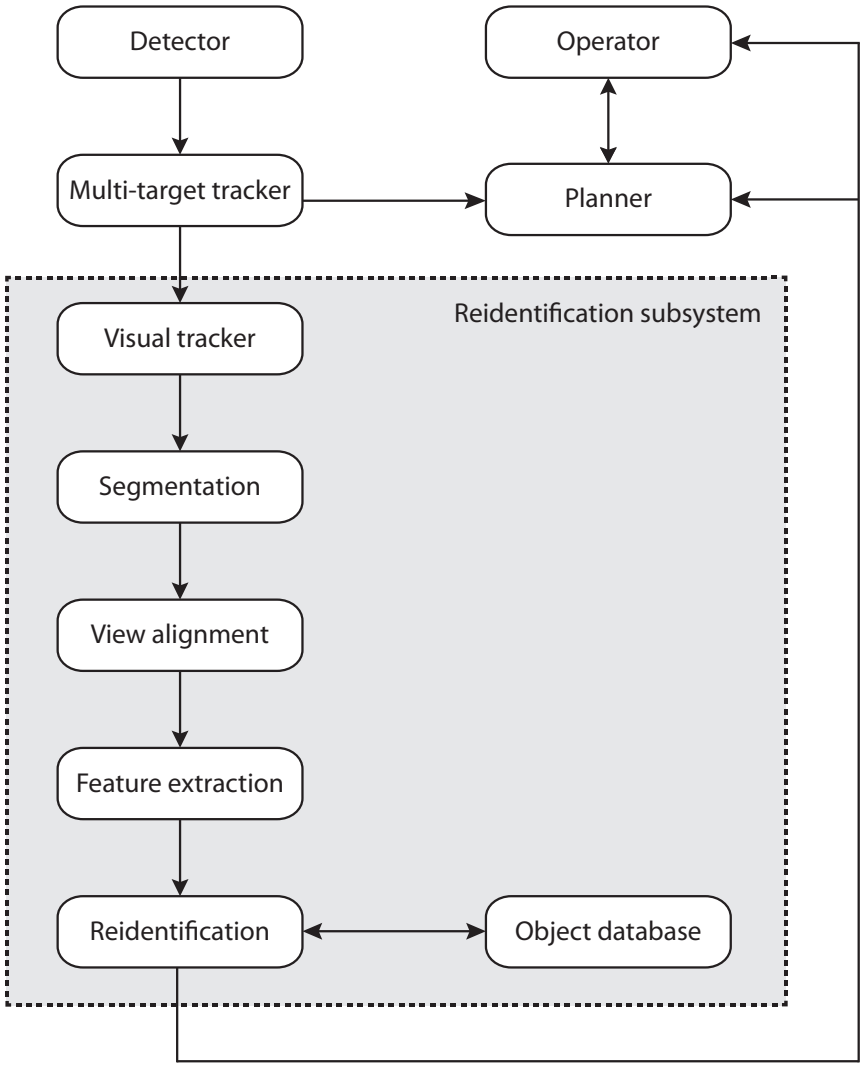


Figure 1.1: Overview of the framework for reidentification of ground targets.

View alignment We have already mentioned the necessity of using distortion tolerant features in the matching process due to the signal data variations described in section 1.3.1. However, there is a contradiction between discriminative power of features, and robustness against signal data variations. Consequently, one should always try to compensate for pose differences between the test and reference images. This compensation step can be referred to as *view alignment*. Estimates of the relative orientation and distance are produced by the multi-target tracker. Automatic view alignment is, however, not yet implemented in our system.

Feature extraction This step extracts robust features that are tolerant against signal variation, but still have discriminative capability. Typical features are SURF, SIFT or color histograms, as described in section 5.3.1.

Reidentification The reidentification step defines the metric between earlier registered instances of detected object and new detections. The output is either "This is a new target, and a new target is added to the target database" or "This is an old target,

and the object signature database should be updated". Comparison methods are given in section 4.3 and 5.2.1.

Object database The purpose of the object database is to gather relevant spatial information of detected ground targets in such a way that new imagery can easily be matched against earlier registrations of detected targets. The structure of as well as the methods for updating such a database are both relevant research areas. Presently, the database consists of subimages (image patches) containing detected targets in combination with pose and distance estimates.

Operator The operator is assumed to interact with the system by issuing high level commands, such as "monitor all activity at place A and B" or "perform autonomous surveillance of road C". The operator can prioritize between different targets, without the need to directly interact with the control of the sensor or platform. This work is not described in this report.

Planner A particle filter based method for simultaneous optimisation of platform trajectory and sensor control is being developed within the project. This work is not described in this report.

2 HLA integration

2.1 Introduction

One important goal has been interoperability with the simulation system in use at SAAB. Their system uses HLA (High Level Architechture for communication between the different modules (or *federates* in HLA terms), e.g. simulation platform, sensor image renderer, target detection, tracking and reidentification. Several HLA federates have been developed at FOI within this and other projects such as SE-MARK and MOSART. There is of course a strong interdependence between these federates. Detection requires simulation of sensor data, tracking requires detection etc. Reidentification relies on several other federates; at least sensor simulation, detection and tracking are required. This chapter introduces the reidentification federate and describes its communication with other federates. First, however, the other federates which are necessary for reidentification are briefly introduced.

2.2 Other federates

2.2.1 Simulation platform

The HLA environment at FOI uses NetScene to keep track of the simulated world, including vehicles (cars, tanks, UAV:s etc.) and sensors. In this context, sensors typically refer to imaging devices either in the visual or thermal infrared spectrum, but any kind of sensor can be used. NetScene provides a user interface for designing scenarios, i.e. defining vehicles and trajectories for them to move along. However, perhaps more importantly, it is also possible for other federates to send HLA messages (called *interactions*) to NetScene, in order to control the vehicles and sensors. This is very useful since it allows feedback within a scenario. For example, a planner federate may respond to an observation (communicated by the detection federate) by planning a new trajectory for a UAV carrying a sensor. So far, however, static scenarios have been used.

Figure 2.1 shows the NetScene application. The large panel in the center shows a map with objects and trajectories, while the surrounding panels provide an interface for changing object properties etc.

2.2.2 Sensor simulation

SceneServer [4] is used to simulate data acquired by visual and infrared sensors. This federate subscribes to information about the current location of all vehicles, as well as sensor parameters such as position, orientation and field of view. Using this information, images are rendered and sent (as HLA interactions) to all federates which subscribe to image data. The interval between images depends on the chosen camera frame rate. In our scenario, the detection federate uses infrared images, while the reidentification federate uses visual images. A typical image is shown in figure 2.2.

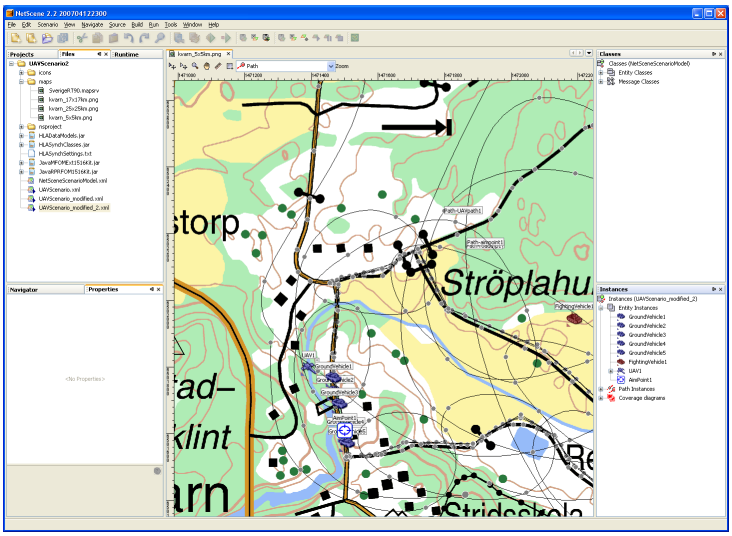


Figure 2.1: A screenshot showing the NetScene application.

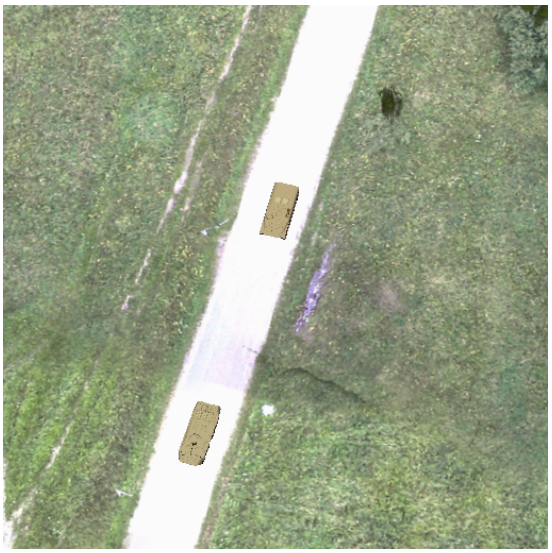


Figure 2.2: A typical simulated visual image from a sensor carried by a UAV.

2.2.3 Detection

At present, there are two detection federates: one real detector which uses infrared images (based on the detection method presented in [12] and one pseudo-detector. The latter "cheats" by subscribing to vehicle locations and calculating their sensor projections. The pseudo-detector is of course very useful when testing other federates, since its detection performance is optimal. However, the real detector has also been shown to provide very good detection performance.

2.2.4 Tracking

A multi-target multi-hypothesis tracker is used for target tracking. The tracker used within this project is highly modular, and hence different process and measurement models can be combined with several filtering and data association techniques. Available filtering modules include extended and unscented Kalman filtering as well as

particle filtering, while data association can be performed using either global nearest neighbors (GNN) or joint probabilistic data association (JPDA). The modular structure also simplifies implementation of additional techniques.

This federate subscribes to observations from the detector and uses them to predict the location of all vehicles. While the observations from the detector tells us that something has been spotted at a certain position, the tracker ties these observations together over time, assigning a unique identifier to each visible vehicle. However, when a vehicle which has not been visible for some time reappears, the tracker is generally not able to associate it with a previous track. This is where reidentification is useful.

2.3 Reidentification

The purpose of the reidentification federate is to resolve the ambiguity which arises when a vehicle, which has not been seen during the last few frames, is detected. By inspecting the visual appearance of the vehicle, the reidentification federate tries to determine if the vehicle is identical to any previously tracked vehicle, or if it has never been observed before. In order to accomplish this, the reidentification federate uses visual images from SceneServer as well as observations from the detector and track predictions from the tracker. The visual information is of course used to recognize the vehicles, while the observations and track predictions are used to determine which parts of the images contain vehicles to be identified.

In the current version of the reidentification federate, the appearance of a vehicle is represented by a color co-occurrence matrix (CCM, described in section 4.2). However, the federate is modular in the sense that the feature representation and matching can easily be replaced without affecting other parts of the federate.

Under ideal circumstances, all vehicles which are visible in any image are observed by the detector. The detector, however, only provides an approximate localization. In particular, the observation does not delineate the vehicle from the surrounding background. This makes extraction of visual features difficult, since we obviously do not want the model of the vehicle's appearance to be affected by the background. In order to overcome this problem, we use a sequence of 10–15 consecutive images where the vehicle is visible. Assuming that the vehicle is moving, its motion relative to the background can be used to obtain an approximate vehicle segmentation. In conjunction with the actual sensor images, this segmentation is used to extract a representation of the vehicle appearance. The segmentation is described in greater detail in chapter 3. In this context, it is sufficient to know that the necessary input to the segmentation algorithm is a sequence of images where a certain vehicle is visible, and a set of image coordinates pointing to the approximate vehicle position in those images.

At a first glance, finding the necessary input parameters to the segmentation algorithm seems like a trivial task. The image sequence is simply a sequence of images from the sensor, and the coordinates are available from the detection federate. Unfortunately, the process is complicated by the fact that any image may contain more than one vehicle, in which case there will be more than one observation from the detection federate. Also, the detection federate may provide false alarms (i.e. incorrect detections), or it may not detect the vehicle at all in some images. All these cases have to be handled in order to provide meaningful input to the segmentation algorithm. A possible solution to these problems would be to attempt to track the observations, associating observations in one image with those in the next and so on. This functionality, however, is performed by the tracker, and should not be duplicated in the reidentification federate. Instead, the reidentification relies both on observations from the detection federate and on track predictions from the tracking federate. These are combined into unambiguous short-term vehicle tracks (referred to as internal tracks

or itracks), containing the coordinates needed to segment the images. Algorithm 1 describes how this is performed.

```
for each image (from sensor simulation) do
    for each track t (from tracker) corresponding to this image do
        find unambiguous observation o (from detector) related to t[a];
        if such an observation was found then
            find itrack i associated with t[b];
            if such an itrack was found then
                append position of o to i[c];
                mark i as updated at current time;
            else
                create new itrack i_new;
                associate i_new with t;
                append position of o to i_new;
                mark i_new as updated at current time;
                append i_new to list of itracks;
            end
        end
    end
    for each itrack i do
        if i is sufficiently long[d] and not previously identified then
            perform segmentation of images using coordinates in i;
            perform reidentification using segmented images[e];
        end
        if i not updated at current time then
            remove i from list of itracks;
        end
    end
end
```

Algorithm 1: Association of observations to tracks, for management of internal tracks.

[a] A match between a track and an observation is considered unambiguous if the predicted track position is close to the observation position, and no other tracks or observations are located near that position.

[b] Each internal track (itrack) is associated with the persistent identifier of one track. Hence the reidentification federate does not need to associate observations at different times with each other, but can instead rely on associations from the tracker.

[c] Each internal track contains a list of observation coordinates, which describe the path of an observed vehicle.

[d] If a vehicle has been seen in at least 10–15 consecutive frames (i.e. if the path stored in an itrack contains at least 10–15 coordinates corresponding to positions in consecutive frames), there is enough information to perform a segmentation.

[e] A representation of the vehicle's appearance is calculated using the segmented image sequence and the internal track. This representation is then matched to vehicles in the object database.

This implementation places rather high demands on the detection federate. Since a vehicle must be visible and observed in a number of consecutive frames in order to be segmented, no vehicles can be recognized unless the detector consistently observes them during that number of frames. This requirement can be relaxed by "trusting" the tracking federate to a greater degree, using the track predictions when no observation is available. This is likely to improve the overall performance of the reidentification federate while also obviating the rather complicated handling of internal tracks. This has not yet been tested, however.

Whenever a vehicle has been observed in a sufficient number of consecutive frames, it is segmented and a representation of its appearance is calculated. The representation is then compared to those of all previously seen vehicles (the details of the matching

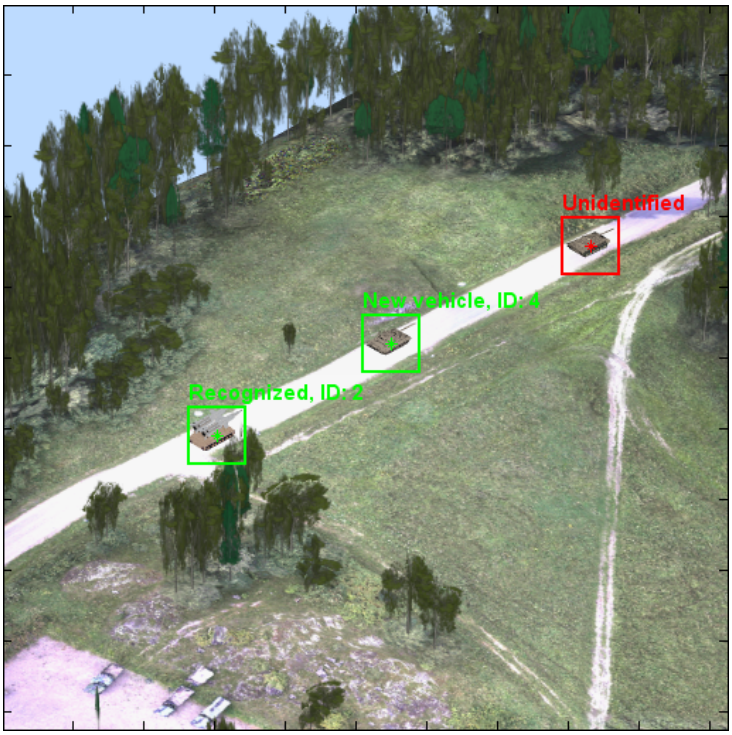


Figure 2.3: The reidentification federate. The status of each vehicle is overlaid on the sensor image.

process are available in section 4.3 and chapter 5 for the two representations discussed in this report), and a matching score is calculated for each vehicle. If the current vehicle is similar to one or more previously seen vehicles, a HLA interaction containing a list of those vehicles and their matching scores is sent. If the newly calculated representation is not similar to any of those stored in the object database, it is assumed that the vehicle has not been seen before. A new internal vehicle ID is then created and associated with the representation, and an HLA interaction containing the new ID is sent.

Figure 2.3 shows a sensor image and the state of the reidentification federate. All currently tracked vehicles are marked as either unidentified, new (not previously seen) or recognized.

2.3.1 Future improvements

During the project, a number of possible improvements have been identified. Most of these have not yet been investigated in any detail, but they are expected to significantly improve the reidentification performance.

1. *Include vehicle geometry:* Currently, only features which can be directly extracted from the sensor images are utilized. However, since the camera parameters (position relative to the ground, orientation and focal length) are known, it would be straight-forward to calculate some basic geometric properties of the observed vehicles. For example, the length and width of the vehicles could be used to exclude some erroneous matches from the identification process.

2. *Prior knowledge of vehicle position:* It is possible to obtain predictions of all vehicle positions from the tracking federate. This information could be used as a prior probability of finding a certain vehicle at a certain position, effectively

excluding all vehicles which have recently been seen far from the currently observed location.

3. *View angle dependence:* The current implementation assumes that all vehicles are seen from above, i.e. that the UAV is flying at a rather high altitude. In some scenarios, it may be more reasonable to expect vehicles to sometimes be seen from the front, back or sides. By keeping track of the current viewing angle, multiple appearance representations could be associated with each vehicle, thereby improving recognition performance when the vehicles are seen from different angles.

4. *Object merging:* When a vehicle is not recognized, it is assigned a new ID. Over time, this is likely to result in all vehicles being associated with several different ID:s (particularly if the vehicles are seen from different sides; see above). A mechanism for merging these ID:s, either manually by operator intervention, or automatically, should be implemented.

5. *Updating of appearance representations:* When a vehicle is reidentified, its corresponding appearance representation should be updated to reflect the new measurements.

3 Object segmentation

3.1 Introduction

Separating the target object from the background and possible foreground occlusion can be difficult. In our segmentation algorithm we use the assumption that target vehicles are moving which simplifies the object separation.

3.2 Segmentation method

The object segmentation problem is handled as an estimation problem with hidden variables. The segmentation is based on an assumption that the target moves during the sequence and that the appearance to be segmented can be approximated by a plane through the object. The plane is tracked using a smoother based on a tracker previously used with good results [13] [9]. Each pixel on and around the object is supposed to originate from one of three possible sources: the foreground, the background or the target itself. The source for each pixel is treated as hidden variables in the smoother that estimates the trajectory. The trajectory and the segmentation is calculated in an algorithm that is basically an approximated maximum a posteriori (MAP) form of the expectation maximation (EM) algorithm [15].

The algorithm iterates between solving the expectation of which pixel that has the target as source given the estimated trajectory (expectation step). The second step is to maximize the a posteriori probability of the trajectory using a smoother given the segmentation (maximation step). Basically the expectation step calculates the probability for each pixel that it belongs to the classes: foreground, background or target.

$$p(I(i)|pixel_i \subset target) \propto exp((I(i) - I_{tgt}(i))^2/R^2) = w_1 \tag{3.1}$$

$$p(I(i)|pixel_i \subset background) \propto exp((I(i) - I_{bg}(i))^2/R^2) = w_2 \tag{3.2}$$

$$p(I(i)|pixel_i \subset foreground) \propto 0.1 \times w_2 = w_3 \tag{3.3}$$

Here, $I(i)$ is the intensity at pixel i, $I_{bg}(i)$ is the intensity of the background template and $I_{tgt}(i)$ is the intensity recorded for the initial target template. The tuning parameter R can be viewed as a standard deviation of the pixel noise. The final foreground probability is initialized using a subjective 10 percent chance that the segmented background in the initial image, actually belongs to the foreground. In a comparison between an image with the target present and the same area without the target there is no way to separate background and foreground based on differences from the target image. It is assumed that the initial image is reasonably free from foreground occlusion thus the 10 percent is as sound approximation. Since the classes are supposed to be exhaustive the probability should sum to one giving the final segmentation

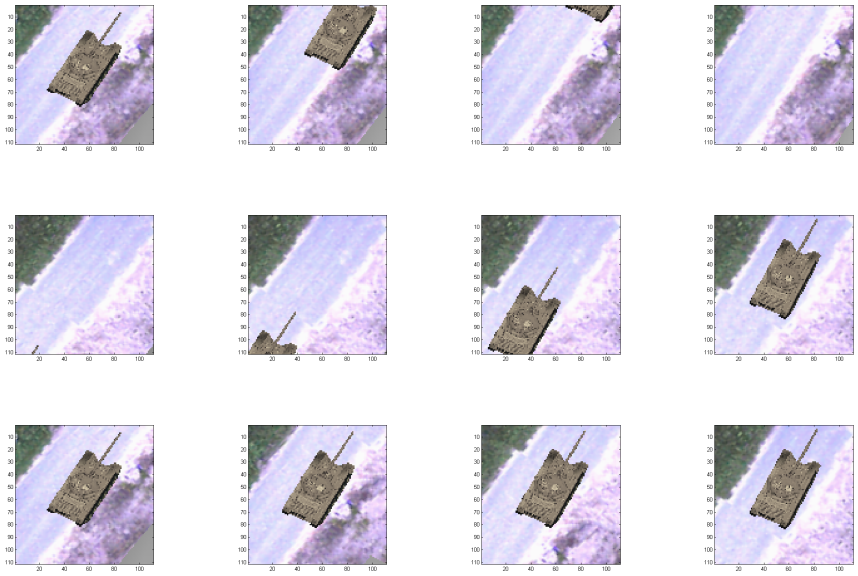


Figure 3.1: Four frames from each of the three sequences of frames that are used by the segmentation algorithm. Each sequence consist of 15 frames. Frame 1, 5, 10 and 15 are shown. The first row is the starting point, the second row the end point and the third row the tracking of the vehicle.

estimates:

$$\hat{p}(pixel_i \subset target) = \frac{w_1}{\sum_{k=1}^{3} w_k} \tag{3.4}$$

$$\hat{p}(pixel_i \subset background) = \frac{w_2}{\sum_{k=1}^{3} w_k} \tag{3.5}$$

$$\hat{p}(pixel_i \subset foreground) = \frac{w_3}{\sum_{k=1}^{3} w_k} \tag{3.6}$$

During the sequence a diffused version of the probabilities from one time is used as an a priori to the next. This allows the foreground template to gain probability in areas covering the target where the image does not match i. e. where a possible occlusion from the foreground can be found. Currently the segmentation algorithm operates on three sub-sequences of an image sequence: one covering the start position of the vehicle through the sequence, one covering the end point and one tracking the vehicle, see figure 3.1. The start and endpoint sequences allow initiation of the background template with a vehicle-free image-patch. The smoother then consists of a fusion of two filters: one working backwards through the image sequence and one working forwards.

3.3 Results and future improvements

The segmentation algorithm shows very promising results for moving vehicles in SceneServer simulations. One such vehicle separation is given in figure 3.2. So far no simulations with partial foreground occlusion of the target vehicles have been performed.

Besides the initialization, the background template is in the current implementation taken from the previous image thus only improving the segmentation where the vehicle has moved. The current implementation has potential for improvements.

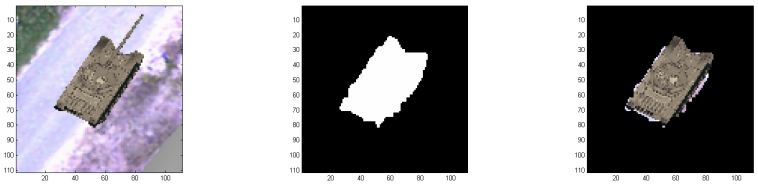


Figure 3.2: Vehicle A before the segmentation, the mask given by the segmentation algorithm and the vehicle after the background has been removed.

One direct improvement would be to include estimation of a vehicle-free background template for each image in the sequence.

4 Color Histogram Matching

4.1 Introduction

Color co-occurrence matrices (CCM) is an fast and simple way for reidentification of objects in images. However, for the CCM to be efficient the objects (for our case target vehicles) need to be segmented from the background, see chapter 3. Therefore our reidentification algorithm first performs an object segmentation and then the identification using CCM. We have been using sensor data within the visual spectrum since CCM is a color matching algorithm. It would also be possible to use CCM in the thermal infrared spectrum, but that does not seem to provide as good recognition performance. It may, however, be beneficial to combine visual and thermal infrared data, or to use multispectral visual images.

CCM was one of the two bases for reidentification that was proposed in the DARPA program VIVID [8] [24] [25]. Our previous work with CCM have been described in [19] [20].

4.2 CCM

Color co-occurrence matrix [3] can be described as a three dimensional color histogram that contains information about the distance between different colors in the image. Since we are using CCM for the visual spectrum we have decided to work within the HSV color space. In this color space the Euclidean distance between colors is approximately proportional to the perceived color difference. Since no spatial information except distance is used to determine CCM, this representation is invariant to image rotation. One CCM is calculated for each color channel and during the matching between two images an intersection is calculated for each channel. This is described in more detail in section 4.3.

Before the CCM is calculated the size the target is estimated based on knowledge about focal length and the distance between the sensor and the target. Thereafter the image is rescaled so that the same target will be approximately the same number of pixels in size each time we observe it. This is done since CCM has a limited invariance to scale.

To limit the size of the CCM and make it robust and simple to handle, the number of colors are quantized to n. The number of distances is given by m. The CCM is then a three dimensional matrix of size $n \times n \times m$. An element $[i, j, k]$ in the CCM describes how often colors i and j occur at distance k from each other in the image. Below we will give some more detailed information about the information stored in a CCM. The colors i, j are labelled $0, 1, ..., n-1$ and the distances $k = 0, 1, ..., m-1$ are given in pixels. For a given distance k we call the $n \times n$ matrix $[0 : n-1, 0 : n-1, k]$ a slice. Each such slice in the CCM is symmetric. The CCM can be describes as consisting of these five parts:

1. ***The matrix*** $[0..n-1, 0..n-1, 0]$***:*** The slice given by the distance $k = 0$ is diagonal and an ordinary color histogram, i.e. the element $[i, i, 0]$ represents the number of pixels with value i in the object.

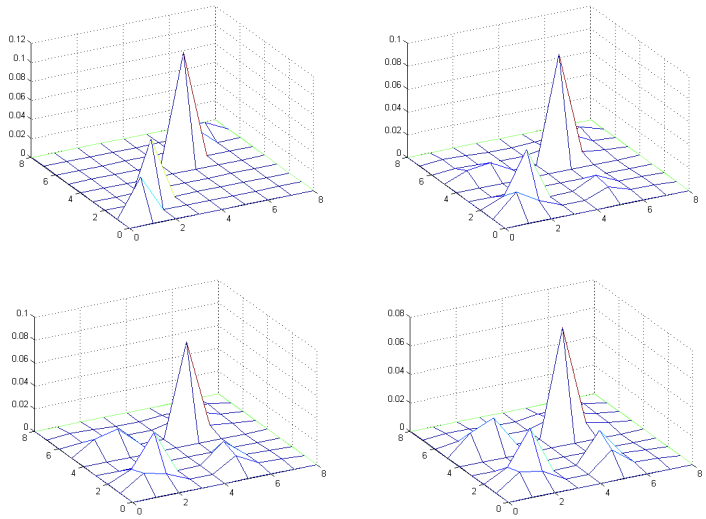


Figure 4.1: Four slices of a CCM for distance $k = 0, 1, 2$ and 3 pixels. The CCM comes from vehicle C.

2. ***The diagonal part of*** $[0..n-1, 0..n-1, 1]$***:*** For the distance $k = 1$ the diagonal elements $[i, i, 1]$ represent the number of pairs of neigbouring pixels of color i. If for example one color x is located as one large connected region in the object the element $[x, x, 1]$ will be large, while for a color y that is spread with few neighboring pixels of the same color the element $[y, y, 1]$ will be small.

3. ***The non-diagonal part of*** $[0..n-1, 0..n-1, 1]$***:*** For the distance $k = 1$ the non-diagonal elements $[i, j, 1]$ with $i \neq j$ represent the number of neigbouring pixels of colors i and j. If for example colors x and y are often located beside each other the element $[x, y, 1]$ is large. Two colors u and v that almost never lie beside each other will give a small value of element $[u, v, 1]$.

4. ***The diagonal part of*** $[0..n-1, 0..n-1, 2..m-1]$***:*** In the remaining slices the diagonal elements $[i, i, k]$ with $k \geq 2$ represent the number of times that color i recurs at distance k. This may either represent a pattern with specific frequency or a large region of uniform color.

5. ***The non-diagonal part of*** $[0..n-1, 0..n-1, 2..m-1]$***:*** The non-diagonal elements $[i, j, 1]$ with $i \neq j$ and $k \geq 2$ represent the number of pixel pairs with colors i and j that occur at distance k.

Figure 4.1 illustrates the four slices given by an $8 \times 8 \times 4$ CCM. Color information is mostly given by the top left plot (representing distance $k = 0$) while texture information can be found in higher order slices.

4.3 Matching

Two target images are matched by taking the intersection I between their CCMs

$$I = \sum_{k=0}^{m-1} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} min(qCCM_{ij}^{k}, lCCM_{ij}^{k}) \tag{4.1}$$

with $qCCM$ being the color co-occurrence matrix from the query image and $lCCM$ being one of the stored target CCM in the database. During the intersection a weighting can be used to control how much of the different slices and their diagonal and

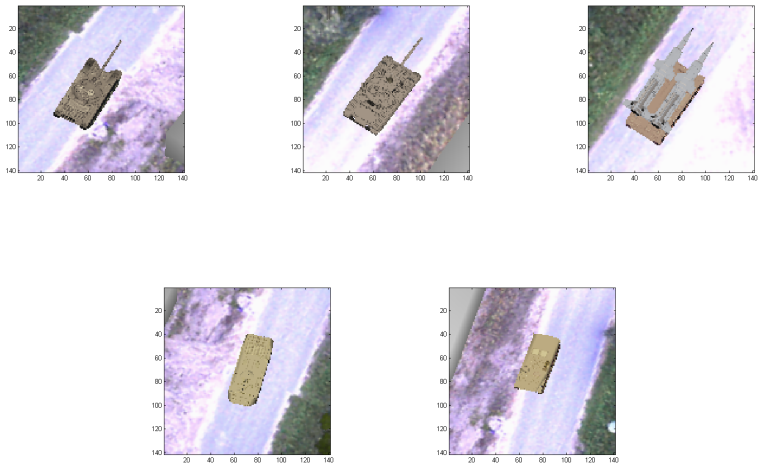


Figure 4.2: The five target vehicles A, B, C, D and E as seen from the UAV in a simulation in SceneServer. The first row illustrates vehicle A, B and C driving from the camera and the second row is vehicle D and E driving towards the camera.

non-diagonal parts should affect the outcome. For color images each object has one CCM for every color channel so the result from these intersections needs to be combined to finalize the matching.

4.4 Results

We have used a scenario with five different vehicles driving in a spread column and an observing UAV that travels back and forth between them. Since the five target vehicles are separated by some distance, the UAV will not be able to observe all of them at the same time. Usually only one or two targets are within the observing range. The five target vehicles A-E can be seen in figure 4.2. The current version of SceneServer does not support antialiasing, cast shadows, and some other advanced rendering options, and hence the produced images are not quite photo-realistic. This impacts the realism of our simulations. At the time of the test, the tracker was still under development and therefore the algorithm for associating observations and tracks could not be used. Instead, exact vehicle localizations were obtained using the pseudo-detector.

Our initial reidentification method based on CCM will give three possible outcomes after the intersection between the query target and the saved targets in the database. A threshold is used to determine whether the CCMs originates from the same object or not.

1. ***No match:*** None of the target CCMs saved in the database is close enough for the query target CCM. Therefore we update the target database with the query target as a new vehicle type.

2. ***One match:*** The query target CCM is close to one of the target CCMs in the database. The query target is labeled with that targets database id. In this initial test the CCM from the matched target in the database is not updated with new information from the query (this could be done for example by taking a weighted mean of the two CCMs).

3. ***More than one match:*** The query target CCM is close to two or more target CCMs in the database. Therefore the query target is labeled with all of these database id's.

Observation	Targer	Labeled as
1	A	not enough data for segmentation
2	A	**1**
3	B	1
4	C	**2**
5	B	**3**
6	A	1, 3
7	A	3
8	B	3
9	C	**4**
10	D	**5**
11	E	5
12	D	5
13	C	**6**
14	B	3
15	A	1, 3
16	A	1
17	B	3
18	C	4
19	D	5
20	E	**7**
21	D	5
22	C	2, 4
23	B	3
24	A	3

Table 4.1: Result after scenario with 24 observations. For each color channel $n = 8$ colors are used and the number of distances $m = 4$. The threshold in the matching is set to 2.25. Each time a new vehicle is labeled it is written in bold.

If the threshold is set too high there will often be a *"no match"* and if the threshold is too low there will often be *"more than one match"*. Therefore a correct threshold is very important.

In table 4.1 we can see the result after one such scenario with 24 observations. The five vehicles are called A, B, C, D and E. The target database was empty at the beginning of the scenario. The first observation was too short for a segmentation to be done and therefore no CCM was calculated. For the second observation a CCM was calculated and since the database was empty the target was labeled *vehicle 1*. The third observation was also labeled *vehicle 1* even though it was not target A but B that was observed. Since target A and B are similar (see figure 4.2) this does not seem too strange. The fourth observation is correctly labeled as the new *vehicle 2*. The fifth observation is also labeled as a new vehicle even though target B have already been observe once before. For the sixth observation two possible vehicles are labeled (*vehicle 1* & *3*). Again, this is not strange, since these vehicles are simular.

In table 4.2 the matching from the scenario is collected. We note that target C is often seen as a new vehicle (*2, 4* & *6*). Target D and E are once mistaken as being the same vehicle but since they are similar even for the human eye that is a reasonable misclassification. As mentioned before the similar looking targets A and B are sometimes mistaken for one another. Positive is that targets A and B are never misstaken for being any other than *vehicle 1* and *3*, target C is the sole target associated with *vehicle 2*, *vehicle 4* and *6*, and finally that target D and E are not associated with any other vehicles than *5* and *7*. So no cross-associations are made between the three target groups [A,B], [C] and [D,E]. By only using a constant threshold and a simple intersection without any weights a promising result has been achieved. Further

Target	1	2	3	4	5	6	7
A	**2**/2		2/2				
B	1		**5**				
C		**1**/1		**2**/1		**1**	
D					**4**		
E					1		**1**

Table 4.2: The collected matching from the 24 observations in the scenario. Bold numbers indicate what target was the first to label that vehicle id. Numbers after a slash comes from observations with more than one match.

optimization of the parameters is expected to improve the reidentification.

5 Boosted distance measurement

5.1 Introduction

In this report we are concerned with methods answering the question "Is this the same object (instance) that was previously observed?". While this may be regarded as a special case of object identification, we prefer to use the term *reidentification*, which is well-established in the field of traffic surveillance.

Reidentification differs from general object recognition (categorization and identification) in that, typically, in the former only a single previous (reference) observation is available of the object of interest, whereas in the latter it is often assumed that a comprehensive signature database can be assembled, covering all possible appearances of the object. Since, *e.g.*, object pose and illumination conditions may be significantly different when the object is observed again, matching methods used in reidentification must be highly robust to such variations.

The observations used in reidentification with data from imaging sensors may be single images or sequences. Consequently, matching may involve two images, two sequences, or one image and one sequence. The data may be generated by the same sensor or two different ones, which may even be of different type (*e.g.*, visual and infrared). In this report we limit ourselves to matching between two images from sensors of the same type.

Reidentification is basically a classification problem. Let $\mathbf{I}_1 \in I_1$ and $\mathbf{I}_2 \in I_2$ denote the images to be compared. We are then looking for a binary-valued function $F : I_1 \times I_2 \rightarrow \{0, 1\}$, such that the output is 1 if the images depict the same object, and 0 otherwise. Sometimes it is instead of interest to have F generate an estimate of the (posterior) probability that the images depict the same object; this is the approach taken in the present work.

We assume objects of the correct category can be detected, manually or automatically, and that a coarse estimate of the object pose can be obtained in each image. Consequently, we also assume the images can be fairly well aligned, though not necessarily on a pixel level.

5.2 Theory

5.2.1 Distance measures

A digital image with N pixels can be represented in vector form $\mathbf{I} = \begin{bmatrix} I_1, I_2, \ldots, I_N \end{bmatrix}^T$. Consequently, two images $\mathbf{I}_1$ and $\mathbf{I}_2$ of equal size can be compared using the Euclidean distance

$$D_E^2(\mathbf{I}_1, \mathbf{I}_2) = (\mathbf{I}_1 - \mathbf{I}_2)^T(\mathbf{I}_1 - \mathbf{I}_2) \tag{5.1}$$

A somewhat more sophisticated measure is the Mahalanobis distance

$$D_M^2(\mathbf{I}_1, \mathbf{I}_2) = (\mathbf{I}_1 - \mathbf{I}_2)^T\mathbf{M}(\mathbf{I}_1 - \mathbf{I}_2) \tag{5.2}$$

where $\mathbf{M}$ is a symmetric positive semi-definite matrix. This makes it possible to take into account that the appearance of different parts of the object covary, and that the appearance is more stable at certain parts than others.

To achieve sufficient robustness against variations in pose and illumination it is necessary to change image representation. We therefore transform the image by means of a mapping $\xi : I \times \Xi$ from the pixel basis I to a feature space Ξ, and form the feature vector $\xi(\mathbf{I})$. Histogram-based representations have proven robust and discriminative, and have therefore become very popular in recent years. A well-known example is the SIFT descriptor [14]. We can now compute the Mahalanobis distance between the images in the feature space:

$$D_M^2(\xi(\mathbf{I}_1), \xi(\mathbf{I}_2)) = (\xi(\mathbf{I}_1) - \xi(\mathbf{I}_2))^T \mathbf{M} (\xi(\mathbf{I}_1) - \xi(\mathbf{I}_2)) \tag{5.3}$$

A good metric $\mathbf{M}$ consistently makes the distance large between images of different objects, and small between images of the same object. Several methods for Mahalanobis metric learning from labeled data pairs have been proposed in recent years, *e.g.*, [18, 23]. These and other methods generally produce full-rank matrices. For computational as well as statistical reasons we find it preferable to learn a low-rank (pseudo-) metric. We choose a method rejected by Xing *et al.*, which produces a rank-one metric matrix.

Let the data set be divided into pairs $Z_p = \{\mathbf{I}_{1,i}^p, \mathbf{I}_{2,i}^p\}_{i=1}^{N_p}$ of images of the same object, and pairs $Z_n = \{\mathbf{I}_{1,i}^n, \mathbf{I}_{2,i}^n\}_{i=1}^{N_n}$ of images of different objects. Here p and n denote positives and negatives, respectively. Furthermore, let $\mathbf{\Delta}\xi_i^p = \xi(\mathbf{I}_{1,i}^p) - \xi(\mathbf{I}_{2,i}^p)$ and $\mathbf{\Delta}\xi_i^n = \xi(\mathbf{I}_{1,i}^n) - \xi(\mathbf{I}_{2,i}^n)$.

We then choose $\mathbf{M}$ such that the average (squared) distance in feature space for positive pairs is minimised, subject to the average distance for negatives being larger than a constant (which may be set to one), *i.e.*,

$$\min_{\mathbf{M}} \sum_{i=1}^{N_p} (\mathbf{\Delta}\xi_i^p)^T \mathbf{M} \mathbf{\Delta}\xi_i^p \tag{5.4}$$

subject to

$$\sum_{i=1}^{N_n} (\mathbf{\Delta}\xi_i^n)^T \mathbf{M} \mathbf{\Delta}\xi_i^n \geq 1 \tag{5.5}$$

$$\mathbf{M} \succeq 0 \tag{5.6}$$

The optimal $\mathbf{M}$ is given by $\mathbf{M} = \mathbf{m}\mathbf{m}^T$, where $\mathbf{m}$ is the eigenvector corresponding to the largest eigenvalue of the generalised eigenvalue problem

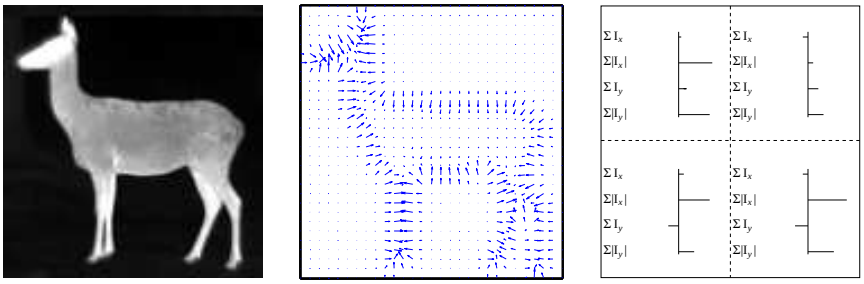
$$\sum_{i=1}^{N_n} \mathbf{\Delta}\xi_i^n (\mathbf{\Delta}\xi_i^n)^T \mathbf{m} = \lambda \sum_{i=1}^{N_p} \mathbf{\Delta}\xi_i^p (\mathbf{\Delta}\xi_i^p)^T \mathbf{m} \tag{5.7}$$

To increase the dimensionality of $\mathbf{M}$, if so desired, we propose using Friedman's *structure removal* [7] technique. Briefly, in the present case, this amount to transforming the feature vector differences $\mathbf{\Delta}\xi$ such that the projection of positive and negative examples onto $\mathbf{m}$, separately, become standard normal distributed, leaving orthogonal dimensions unchanged. This effectively removes any separation between the classes in the direction $\mathbf{m}$, and forces the optimisation algorithm to select a complementary direction. The procedure is suboptimal, since removing structure in one direction may re-introduce class separation in a previously "removed" direction. Structure removal has nevertheless proven useful in increasing the number of projections in linear discriminant analysis [1].

5.2.2 Features

The SURF descriptor

We have already emphasized the importance of using robust discriminative features in order to achieve tolerance to variations in pose and illumination. Based on positive



a b c

Figure 5.1: SURF descriptor. (a) Thermal infrared image of a deer. (b) Gradient image. (c) The image (region) is divided into a set of subregions (here $2 \times 2 = 4$). In each subregion the sum of filter responses and their absolute values for derivative operators in horizontal and vertical directions are computed. The final descriptor is obtained by concatenating the components and normalising the result.

experience from previous investigations in object detection, we have chosen to use the *SURF* descriptor [2] in the present work. SURF is similar to the SIFT descriptor in using a gradient image as source. But instead of forming histograms of gradient vector directions in rectangular image subregions, the SURF descriptor represents a subregion $\mathcal{R}$ as a four-vector

$$\sum_{(x,y)\in\mathcal{R}} \left[I_x, |I_x|, I_y, |I_y|\right]^T \tag{5.8}$$

of spatial sums of partial derivatives and their absolute values. The complete descriptor for an image region is obtained by concatenating the descriptors of the subregions, and normalising the resulting vector. See Figure 5.1.

Colour histogram

SURF, of course, has a fundamental limitation in that it (like SIFT and other similar constructs) only describes shape. As a results, objects of similar shape but with different colours may be falsely classified as equal. To add colour discrimination ability we use a histogram-based colour descriptor proposed in [21].

For every subregion used in assembling the SURF descriptor a colour histogram is computed, where the colour space is divided into a set of colour classes, based on colour names assigned by humans to example data. The basic idea is to achieve a certain tolerance to photometric variations, without losing as much discriminative power as when using photometric invariants.

We use eight colour names: black, blue, brown, green, grey, red, white, and yellow. These are the dominant colour names for cars according to DuPont's annual Color Popularity Report of vehicle colours [1]. We collected 400 samples of car colours from 21 manufacturers' websites ("Build your car"). Only colour samples that had names containing any of the eight colour names were used (with exception for "silver", which was interpreted as white or grey depending on the average intensity). The colour samples where transformed from RGB to the more perceptually uniform L*a*b* representation. For each colour class this space was divided into $256 \times 256 \times 256$ bins and the pixels from the colour samples in each class were distributed among these. The data volumes were subsequently reduced via lowpass filtering and subsampling into $8 \times 16 \times 16$ bins. After normalisation this gives an estimate of the probability density

[1] `http://www.dupont.com`

$p(\text{L*a*b}|c_k)$ for observing a particular L*a*b combination in colour name class c_k. Using Bayes' theorem

$$P(c_k|\text{L*a*b}) = \frac{p(\text{L*a*b}|c_k)P(c_k)}{\sum_l p(\text{L*a*b}|c_l)P(c_l)} \tag{5.9}$$

posterior probabilities for each colour class can be computed at each pixel position. We average the posterior colour probabilities in each SURF subregion and form an eight-vector. The complete colour descriptor is obtained by concatenating the feature vectors from all subregions and normalising the result. Finally, the colour descriptor is concatenated to the SURF vector to form a combined description of colour and shape.

5.2.3 Boosting

Reidentification is a classification problem, and the methods presented in Section 5.2.1 makes it possible to design a simple classifier by determining, from training data, a threshold value for the distance between two images, such that if the distance is larger than the threshold, the images are deemed to depict different object. However, we can achieve much better results by combining a number of local distance measurements.

We use LogitBoost [10] to model the probability that the images depict the same object. LogitBoost (and, incidentally, AdaBoost) is a procedure for determining, from training data, the exponent F in the logistic model

$$P(\text{same}|\mathbf{I}_1, \mathbf{I}_2) = \frac{1}{1 + \exp(-F(\mathbf{I}_1, \mathbf{I}_2))} \tag{5.10}$$

LogitBoost does this by an iterative procedure $F^k = F^{k-1} + f_k$, where in each iteration the current estimate is modified by determining a function f that satisfies

$$L(F^{k-1} + f) \geq L(F^{k-1}) \tag{5.11}$$

where L is the binomial likelihood function for the probability model. L is concave in F so there is a single extremum. f is computed as an approximate Newton-Raphson step from the current position in function space. This, however, specifies f only for arguments corresponding to the training examples. To be useful, f must be extended to the entire space of image pairs. This is done by defining a function family G and selecting, by means of least-squares fitting, the family member that best approximates the ideal f for training data.

In the present work we choose G to be binary regression trees where, at each internal node, the distance between two corresponding image regions is tested against a threshold. More specifically, we limit ourselves to stumps, *i.e.*, binary trees with a single internal node.

Figure 5.2 shows a pair of images to be compared. In the images are also shown the three first regions selected by the boosting algorithm. At each boosting stage a large number of candidate regions of random size and position are generated, and the algorithm picks the region whose associated regression stump best approximates the current ideal boosting step f. If we use multiple feature types, we randomly select a representation for each region.

To conclude, the boosting algorithm generates a discriminant function

$$F(\mathbf{I}_1, \mathbf{I}_2) = \sum_k f_k(\mathbf{R}_{1k}, \mathbf{R}_{2k}) \tag{5.12}$$

where f_k is a regression stump and $\mathbf{R}_{1k}$ and $\mathbf{R}_{2k}$ are corresponding image regions in the two images. The stump computes

$$f_k(\mathbf{R}_{1k}, \mathbf{R}_{2k}) = \begin{cases} f_k^1, & D_{M_k}(\xi_k(\mathbf{R}_{1k}), \xi_k(\mathbf{R}_{2k})) < \tau_k \\ f_k^2, & D_{M_k}(\xi_k(\mathbf{R}_{1k}), \xi_k(\mathbf{R}_{2k})) \geq \tau_k \end{cases} \tag{5.13}$$

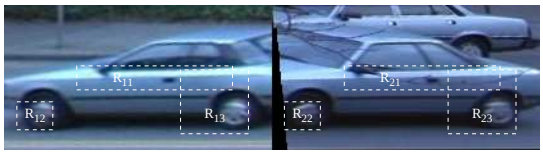


Figure 5.2: Three image regions selected by boosting.

(a)

(b)

Figure 5.3: Examples of data used by Ferencz *et al.*(a) Original imagery. (b) Geometrically transformed images.

where D_{M_k} is the Mahalanobis metric optimised for image region k, ξ_k the selected feature type, and f_k^1, f_k^2, and τ_k are parameters determined by the training process. If the discriminant sum F is greater than a threshold value, the image pair is classified as depicting the same object.

5.3 Results

We have applied our reidentification algorithm to public data generated by Ferencz *et al.*[5]. This data set consists of 4162 images of 358 unique cars. Every car is depicted from two different views, with a small number of images for each view, see Figure 5.3(a). The images are subsequently geometrically transformed so that the car side is parallel to the image plane, and the distance between the front and rear wheel is constant, see Figure 5.3(b).

Cross-validation was used when training, *i.e.*, the data set was divided into k disjunct subsets, of which $k-1$ were used for training and one for validation. There are k ways to choose a unique validation subset, and that many training sessions were therefore conducted. From the set of validation results the expected algorithm performance with uncertainty was estimated. We used $k = 5$ in our experiments.

All images of each vehicle were used either for training or validation (*i.e.*, they were not divided between the training and validation subsets). While approximately 20000 positive examples (*i.e.*, pairs of images depicting the same vehicle) can be generated from the data set, the number of unique negative examples that can be generated is much larger. It should be noted, however, that although there are on average about ten images of each vehicle, many of these are similar. Consequently, on should not anticipate classification performance at the level expected for an equal size set of independent training examples.

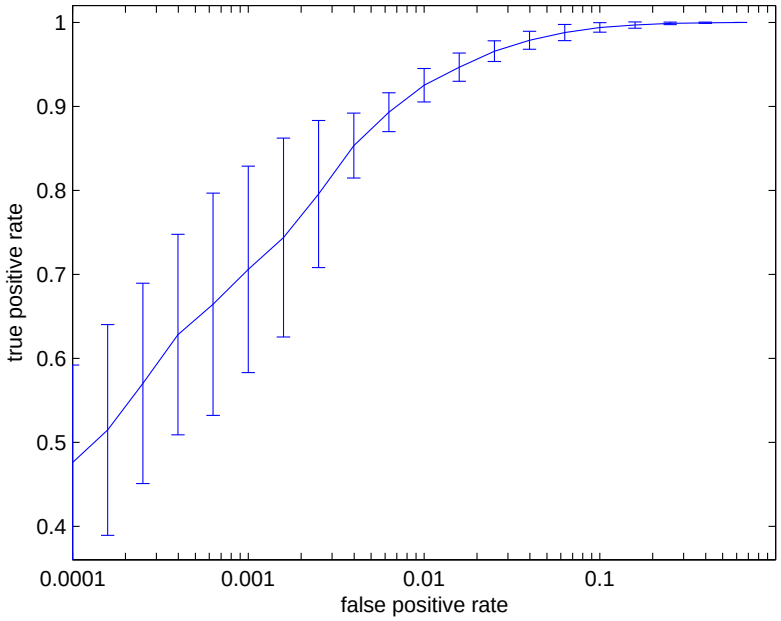


Figure 5.4: ROC curve for reidentification algorithm using 50 SURF features. Uncertainty bounds (one standard deviation) based on five-fold cross-validation are shown.

5.3.1 Validation results

SURF features

We used boosting to combine 50 distance measurements between SURF-represented image regions. At each iteration the algorithm selected one of 50 randomly generated regions. The SURF descriptors used 2×2 subregions with a minimum with of 7 pixels (the distance between the front and rear wheels is 150 pixels). In each cross-validation batch training data consisted of 30000 negatives and approximately 19000 positives from images of about 280 vehicles. The validation used 7500 negatives and approximately 4750 positives from about 70 vehicles. Figure 5.4 shows a ROC-curve with uncertainty bounds for performance on validation data. In Figure 5.5 the equal-error rate is plotted as a function of the number of boosting stages.

Although not directly comparable, we believe our results are better than with previous approaches [6, 11, 16] applied to this data. Furthermore, it is clear from Figure 5.5 that performance can be improved by adding more components.

SURF and colour histograms

In Figure 5.6 is shown a ROC-curve for an algorithm using 50 combined SURF descriptors and colour histograms; all other parameters are as above. The performance is significantly better than without colour information. In Figure 5.7 the equal-error rate is plotted as a function of the number of boosting stages. Note that with just 15 components the performance is already better than with 50 SURF components.

To determine the relative importance of shape and colour information one may study the norm of the part of $\mathbf{m}$ (cf. Section 5.2.1) that acts on the SURF-descriptor (16 components). Figure 5.8 shows a histogram of the norm of this sub-vector of the unit-norm $\mathbf{m}$. Clearly the contribution from the SURF descriptor is generally small, not to say insignificant.

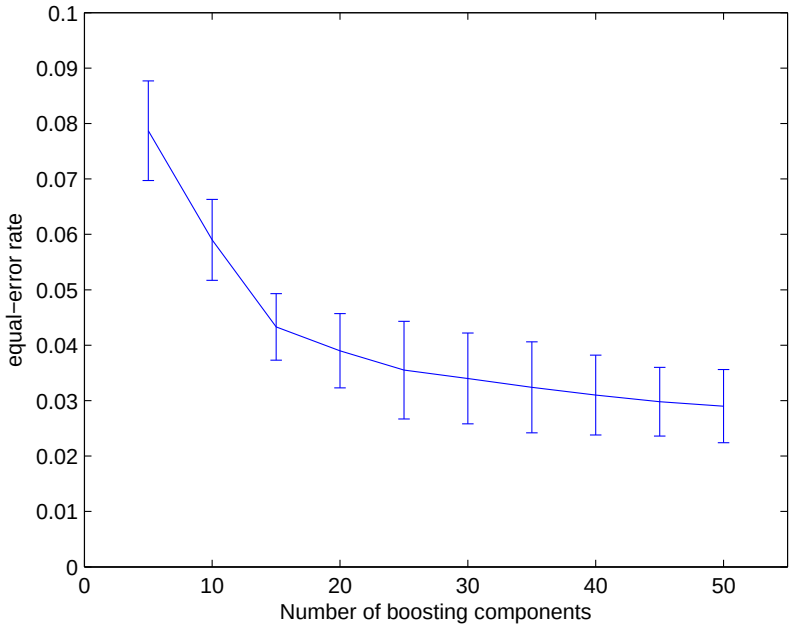


Figure 5.5: Equal-error rate as a function of the number of boosting components (*i.e.*, distance measurements) using SURF features. Error bars show ones standard deviation.

5.4 Discussion

In the present work we have assumed that the images to compare are roughly aligned, and we therefore do not search for best matching regions. We cope with remaining misalignment by using robust distortion-tolerant region descriptors and by learning an optimal region-specific metric. Although we believe this to be a reasonable approach for rigid objects–since a sufficiently accurate pose estimate can often be obtained once the object is detected–it is straightforward to introduce a (correlation type) search phase.

The situation is somewhat different for articulated objects, in particular pedestrians. A natural approach in this case is to use SIFT or similar feature matching techniques to handle significant pose variation. Alternatively, if sequences (of gait cycles, etc.) are available for matching one may search for the best matching images in each sequence using the approach taken in the present study. An exhaustive search may be avoided by first applying a pose estimation algorithm, *e.g.*, [17].

In our work we have used regression stumps to approximate the ideal step in each boosting iteration. It is straightforward to generalise this to arbitrary regression trees by allowing the tree growing algorithm to use more than one image region comparison. A computationally attractive approach may be to build a larger tree for each image region using multiple projections/metrics generated using the structure removal method, as proposed in Section 5.2.1.

The reidentification algorithm is well suited for real-time execution in a single microprocessor or DSP. The extraction of SURF as well as colour histogram region descriptors can be efficiently implemented using the integral image representation [22]. The use of low-rank (pseudo-) metrics also contributes to computational efficiency.

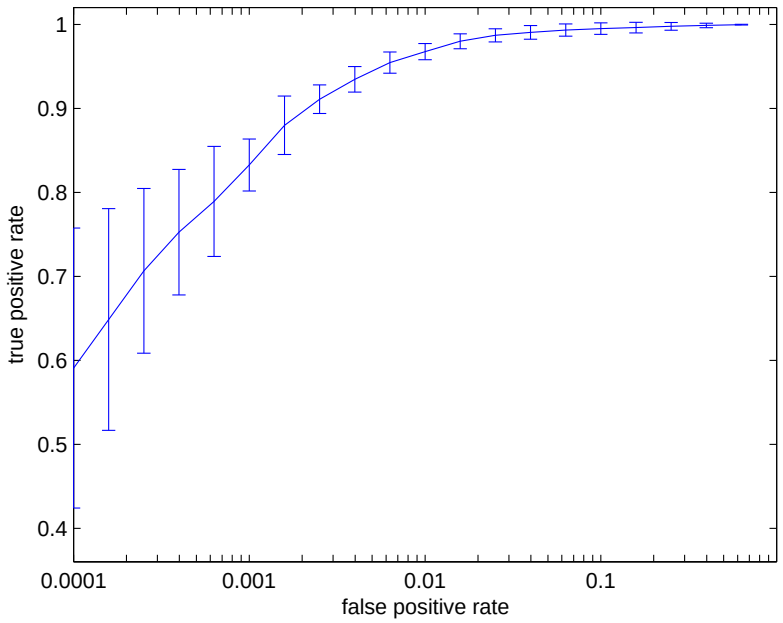


Figure 5.6: ROC curve for reidentification algorithm using 50 combined SURF and colour histogram features. Uncertainty bounds (one standard deviation) based on five-fold cross-validation are shown.

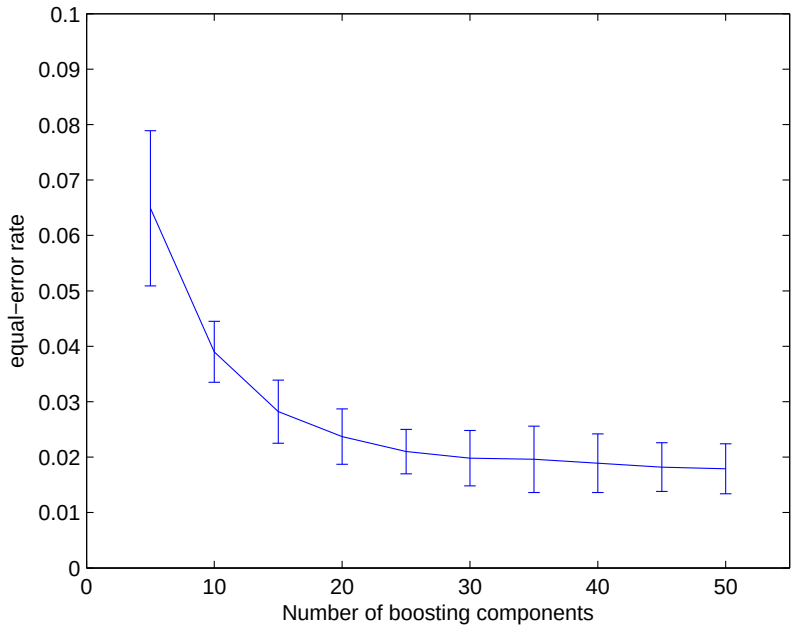


Figure 5.7: Equal-error rate as a function of the number of boosting components (*i.e.*, distance measurements) using combined SURF and colour histogram features. Error bars show ones standard deviation.

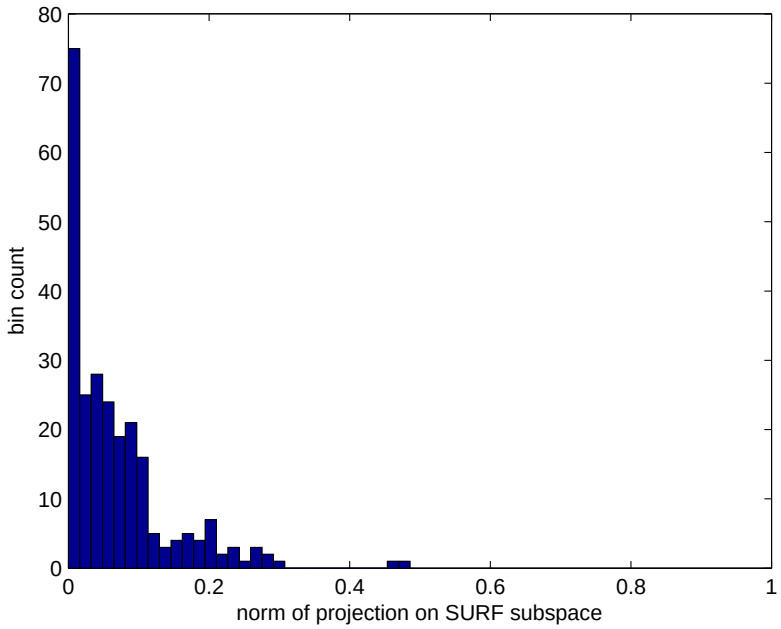


Figure 5.8: Histogram of the norm of the SURF part of the projection vector $\mathbf{m}$.

6 Field trial

6.1 Background

6.1.1 Purpose

The purpose of the data collection was to capture a large number of vehicles in different views from several sensors. The data will be used to benchmark the re-identification process.

6.1.2 Location

The sensors where place on top of the roof at the FOI-building near the parking-lot. During a 3 hour long morning session approximately 100 different vehicles where captured.

6.1.3 Sensor setup

The placement of the sensors was done so we could capture a vehicle in many different views. In order to get as many views of a vehicle as possible from each sensor, we focused our sensors at the entrance of the parking-lot (marked as the red circle). To get into the parking-lot the cars must do a 90 degree turn. During the turn each sensor will see the vehicle from several angles. We also had a secondary setup with a single visual sensor that captures vehicles in a side view (marked as a black circle).

Figure 6.1: Overview of the FOI-building and parking-lot.

Figure 6.2: View from SONY3 (left) and SONY5 (right).

Figure 6.3: View from SONY2 (left) and IR1 (right).

Figure 6.4: View from SONY4 (left) and IR2 (right).

6.2 Sensors used

We used 5 visual standard video cameras, 2 Marlin F-145 cameras in a stereo rig and 2 Thermal infrared cameras (EO/IR). The thermal infrared cameras are placed in pairs with a visual camera.

Model	Sensor Type	Quantity	Name
FLIR QWIP	IR	1	IR1
FLIR SC2000	IR	1	IR2
Sony DCR-DR190	Visual	5	SONY1-5
Marlin F-145 C2	Visual CMOS	2	MARLIN

6.3 Sensor views

Example views from different sensors are shown in figures 6.2 to 6.5.

6.4 Annotation

Each vehicle will be annotated in a couple of different frames from a reference sensor. The annotation will mark the wheelbase of a vehicle and its unique identity. With help of camera-calibration this marking can be transformed to other sensors. The annotation also provides information about the rotation of a vehicle. A special annotation tool has been developed for this task.

Figure 6.5: View from SONY1 (left) and Marlin (right).

Figure 6.6: The wheelbase of a Volvo XC90 is marked with a thin blue line.

7 Conclusion and future work

This report describes the current status of a framework for reidentification of ground targets in aerial surveillance data being developed at FOI. A complex signal processing chain containing the following subparts is discussed: detection, multi-target tracking, visual tracker, segmentation, view alignment, feature extraction, reidentification, object database, planner, and operator interaction.

The main conclusion is that reidentification is a necessity in an autonomous surveillance system, and that promising results have been achieved on simulated scenarios, but also on public databases containing images of civilian cars. The implemented framework still lacks some system functionalities, such as view alignment and a structured aggregation of new imagery in the object database.

7.1 Future work

Field trial database and performance evaluation All sensor data from the field trial will be annotated semi-automatically. This database will be used to measure performance of the method described in chapter 5 over a larger span of poses, and it will also be used for developing method for the object database.

Prior Knowledge Prior knowledge, such as terrain models, vegetation type, and road network, should be used in the multi-target tracker to enhance the prediction of target motion. The modular structure of the multi-target tracker code permits such models, but this has not yet been implemented and evaluated.

Alignment An alignment method has to be implemented to achieve a better pose registration of new detections to object candidates in the database.

Object database Multiple appearance representations (a set of "canonical views") should be implemented to handle the discontinuous nature of signal variation for different poses.

Object merging When a vehicle is not recognized, it is assigned a new ID. Over time, this is likely to result in all vehicles being associated with several different ID:s, particularly if the vehicles are seen from different sides. A mechanism for merging these ID:s, either manually by operator intervention, or automatically, should be implemented. This can be seen as redundancy elimination or pruning of the database, and can be implemented by applying the reidentification algorithm to pairs of database entries.

Operator interaction Operator interaction can enhance the performance in many situations, such as target prioritization, target track merging,

Bibliography

[1] M. Aladjem. Linear discriminant analysis for two classes via removal of classification structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):187–191, 1997.

[2] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In *Proceedings of the 9th European Conference on Computer Vision*, 2006.

[3] P. Chang and J. Krum. Object recognition with color cooccurrence histograms. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

[4] S. Fenelius F. Bennet. Sceneserver - a 3D software assisting developers of computer vision algorithms. Technical Report FOI-R–0831–SE, Swedish Defence Research Agency, Linköping, 2003.

[5] A. Ferencz, E. Learned-Miller, and J. Malik. Learning hyper-features for visual identification. In *Advances in Neural Information Processing Systems 17*, 2005.

[6] A. Ferencz, E. G. Learned-Miller, and J. Malik. Learning to locate informative features for visual identification. *International Journal of Computer Vision*, 77(1-3):3–24, 2008.

[7] J. H. Friedman. Exploratory projection pursuit. *Journal of the American Statistical Association*, 82:249–266, 1987.

[8] D. Guarino, B. Walls, and E. Miles. Confirmatory identification of targets in video - final report. *Video Verification of Identity (VIVID): Automated Video Processing for Unmanned Aircraft - A compilation of scientific papers and technical reports that summarizes the accomplishments of the DARPA VIVID program, Phase 1, Edited T. Strat and L. Hollan*, 2005.

[9] G. D. Hager and P. N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[10] T. Hastie J. Friedman and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

[11] V. Jain, A. Ferencz, and E. Learned-Miller. Discriminative training of hyper-feature models for object identification. In *British machine vision conference*, volume 1, pages 357–366, 2006.

[12] J. Karlholm. Implementering av en detektionsalgoritm för realtidsbearbetning av IR-video. Technical Report FOI-R–1761–SE, Swedish Defence Research Agency, Linköping, 2005.

[13] H. Karlsson, J. Nygårds, and M. Ulvklo. Efficient region tracking and target position estimation in image sequences using Kalman filters. Technical Report FOI-R–0595–SE, Swedish Defence Research Agency, Linköping, September 2002.

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

[15] G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley & Sons, 1996.

[16] E. Nowak and F. Jurie. Learning visual similarity measures for comparing never seen objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[17] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *Proceedings of the International Conference on Computer Vision*, 2003.

[18] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng. Online and batch learning of pseudo-metrics. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[19] K-G. Stenborg. Re-identification of previously observed targets in EO/IR-data from UAV - a survey. Technical Report FOI-R–2335–SE, Swedish Defence Research Agency, Linköping, September 2007.

[20] K-G. Stenborg, M. Ulvklo, and J. Rydell. Implementering och värdering av bildbehandlingsmetoder för återigenkänning. Technical Report FOI-Memo–2268–SE, Swedish Defence Research Agency, Linköping, December 2007.

[21] J. van de Weijer and C. Schmid. Applying color names to image description. In *Proceedings of the International Conference on Image Processing*, 2007.

[22] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[23] E. Xing, A. Y. Ng, M. Jordan, and S. Russell. Distance metric learning, with application to clustering with side -information. In *Advances in Neural Information Processing Systems 15*, 2003.

[24] Z. Yue, D. Guarino, and R. Chellappa. Synthesis of novel views of moving objects airborne video. *Video Verification of Identity (VIVID): Automated Video Processing for Unmanned Aircraft - A compilation of scientific papers and technical reports that summarizes the accomplishments of the DARPA VIVID program, Phase 1, Edited T. Strat and L. Hollan*, 2005.

[25] Z. Yue, D. Guarino, and R. Chellappa. Moving objects verification in airborne video. In *EEE International Conference on Computer Vision Systems (ICVS)*, New York, 2006.