Evaluating Multi-Object Tracking

Kevin Smith, Daniel Gatica-Perez, Jean-Marc Odobez, and Sileye Ba*
IDIAP Research Institute, Martigny, Switzerland
{smith, gatica, odobez, sba}@idiap.ch

Abstract

Multiple object tracking (MOT) is an active and challenging research topic. Many different approaches to the MOT problem exist, yet there is little agreement amongst the community on how to evaluate or compare these methods, and the amount of literature addressing this problem is limited. The goal of this paper is to address this issue by providing a comprehensive approach to the empirical evaluation of tracking performance. To that end, we explore the tracking characteristics important to measure in a real-life application, focusing on configuration (the number and location of objects in a scene) and identification (the consistent labeling of objects over time), and define a set of measures and a protocol to objectively evaluate these characteristics.

1 Introduction

Although object tracking is considered a mature field of research, there is a disturbing lack of uniformity in how results are presented by the community. Award-winning tracking papers rarely use the same data or metrics. This makes comparisons between different methods difficult and stifles progress. At the root of this problem is the lack of common data sets and performance measures (with few exceptions, such as PETS [5]). In this paper, we will outline a framework for evaluating MOT methods which allow for (1) numerical evaluation to measure performance, and (2) visualization to understand tracking phenomena. We will also define specific measures and protocols within this framework relevant to academic and real-life evaluation.

In order to define a framework for tracking evaluation, it is important to understand what qualities are essential to good tracking. To do so, it can be helpful to consider what constitutes a "golden" multi-object tracker. One could argue that a good tracker, in a real-life situation, should:

1. start automatically,
2. *track* objects well - place the correct number of trackers at the correct locations each frame,
3. *identify* objects well - track individual objects consistently over a long period of time,
4. track objects in spite of distraction (occlusion, illumination changes, etc.),
5. accurately estimate task-specific object parameters (such as object velocity),
6. be fast.

This list can be reduced to four key properties (items 2, 3, 5, and 6). Item 2 refers to the *configuration*, or the number and location of objects in the scene. Item 3 refers to *identification*: the consistent labeling of objects over a long period of time. Items 1 and 4 depend on the model and type of tracking algorithm, and can be indirectly measured by measuring the configuration and identification. Item 5 refers to the ability of the method to correctly predict some *task-specific* parameter of an object in the scene. Finally, item 6 refers to the *speed*, or computational cost of the method. This paper focuses on the more generic tasks of evaluating configuration and identification.

There have been recent attempts to measure configuration and identification in the past with various degrees of success [4, 2, 3, 6]. In [4], measures were proposed to evaluate the configuration of a single-object tracker to a limited degree. In [2], configuration and identification were evaluated based on the distance between centroids of the objects and the trackers. However, this method does not account for two differently shaped objects that share a centroid. In [6], configuration measures were defined similar to the four configuration errors we propose here, but an overall configuration measure is not provided and the issue of identification is not addressed. In the very recent work of [3], two identification measures and two configuration measures similar to ours were independently proposed, but with important distinctions, detailed in later sections. Evaluating computational cost can be a complex task [7], and is beyond the scope of this paper.

The remainder of this paper is organized as follows. First, we introduce fundamental concepts to tracking evaluation in Section 2. We describe, in detail, how to evaluate configuration in Section 3. Section 4 describes identification evaluation. Section 5 outlines how task-specific measures can be fit into the framework, and Section 6 contains some final remarks.

*This work was supported by the Swiss National Center of Competence in Research on Interactive Multimodal Information Management (IM2), and by the European Union 6th FWP IST Integrated Project AMI (Augmented Multi-party Interaction, FP6-506811, publication AMI-74).

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)
1063-6919/05 $20.00 © 2005 IEEE

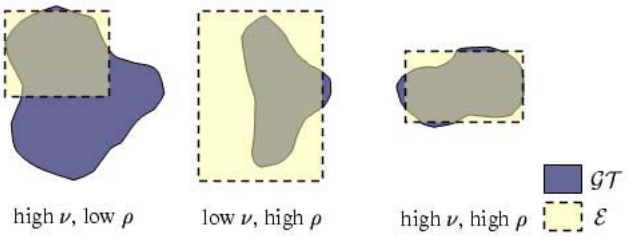

Figure 1: Precision (ν) and recall (ρ). For good quality tracking, both precision and recall should have high values (seen right).

2 Basic Concepts

For this document, objects or tracking targets are referred to as *ground truth objects* ($\mathcal{GT}$), are denoted by lower-case characters (*a, b, c*,...), and indexed by j. Tracker outputs are referred to as *estimates* ($\mathcal{E}$), are represented by numbers (1,2,3,...), and are indexed by i. We make the assumption that $\mathcal{E}$s and $\mathcal{GT}$s can be described as sets of templates with corresponding transformations. For example, a common tracker output is a bounding box with translation and orientation information $\mathcal{E}_i = (x_i, y_i, \theta_i)$.

Two fundamental measures which do not depend on the shape of the $\mathcal{GT}$ or the $\mathcal{E}$ are *precision* and *recall*. They are often used in information retrieval [1], but can also be useful for determining *if* and *how well* an object is being tracked.

Recall. Given a ground truth $\mathcal{GT}_j$ and a tracking estimate $\mathcal{E}_i$, the recall, $\rho_{i,j}$, is expressed as

$$\rho_{i,j} = \frac{|\mathcal{E}_i \cap \mathcal{GT}_j|}{|\mathcal{GT}_j|}. \quad (1)$$

Recall measures how much of the $\mathcal{GT}$ is covered by the $\mathcal{E}$ and can take values between 0 (no overlap) and 1 (fully overlapped). It is possible to have a high recall yet have poor quality tracking, as seen in the center of Figure 1.

Precision. Precision, $\nu_{i,j}$, is defined similarly as

$$\nu_{i,j} = \frac{|\mathcal{E}_i \cap \mathcal{GT}_j|}{|\mathcal{E}_i|}. \quad (2)$$

Precision measures how much of the $\mathcal{E}$ covers the $\mathcal{GT}$ and can take values between 0 (no overlap) and 1 (fully overlapped). It is possible to have high precision with poor quality tracking, as seen on the left of Figure 1.

Coverage Test. The *coverage test* determines if a $\mathcal{GT}$ is being tracked, or if an $\mathcal{E}$ is tracking. We can see from Figure 1 that good tracking requires both high ρ and ν values. The coverage test should only label $\mathcal{E}_i$ as "tracking" or $\mathcal{GT}_j$ as "tracked" if the $\rho_{i,j}$ and $\nu_{i,j}$ values are high. The *F-measure* ($F = \frac{2\nu\rho}{\nu+\rho}$) is suited to this task, as F is only high when both ρ and ν are high [1]. Thus, to determine if $\mathcal{E}_i$ is being tracked by $\mathcal{GT}_j$ and vice-versa, the F-measure must exceed t_C, a *coverage threshold* ($F_{i,j} > t_C$). In the simplest case, $t_C = 0$, so that if there is any overlap between the $\mathcal{E}_i$ and the $\mathcal{GT}_j$, the coverage test is passed.

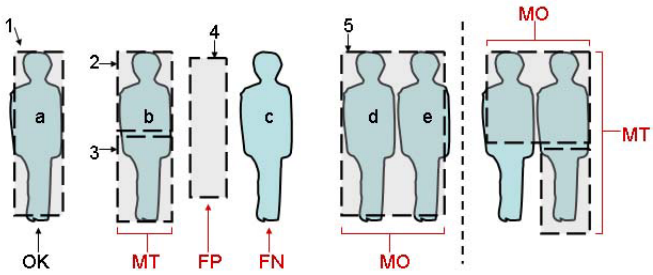

Figure 2: Types of configuration errors. Five bounding boxes $\mathcal{E}$s (*1,2,3,4,5*) attempt to track five $\mathcal{GT}$s (a, b, c, d, e).

3 Configuration

Configuration refers to the number and placement of objects in the scene. In order to evaluate configuration, we must first define what a correct configuration is: a tracker is correctly configured when exactly one $\mathcal{E}$ ($\mathcal{GT}$, resp.) is tracking each $\mathcal{GT}$ ($\mathcal{E}$, resp.). Any departure indicates an error in configuration. An $\mathcal{E}$/$\mathcal{GT}$ is considered to be "tracking/tracked" if it passes the coverage test. In this section, we present five configuration measures (labeled C-1 to C-5) and a procedure for evaluating the configuration.

3.1 Configuration Errors

It can be useful to consider a tracking system whose sole task is to estimate the correct object configuration. Looking at the modes of failure for such a system, it is possible to understand the modes of configuration failure. An object counter, for instance, simply reports the locations and number of objects in the scene without considering identity. There are four basic types of errors that this system can make. First, the system may indicate the presence of an object which does not exist. This type of error is shown in Figure 2 where estimate 4 appears over an empty background. A second type of error may occur when an object exists, but the system does not recognize it (as object c in Figure 2). The third type of error occurs when one object is tracked by multiple estimates (e.g. b is tracked by 2 and 3). The last type of error occurs when multiple objects are tracked by one estimate (e.g. d and e are tracked by 5). Each of these errors corresponds to one of the four types of configuration errors:

- **Measure C-1: (FP)** - False Positive. An estimate exists that is not associated with a ground truth object.
- **Measure C-2: (FN)** - False Negative. A ground truth object exists that is not associated with an estimate.

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)
1063-6919/05 $20.00 © 2005 **IEEE**

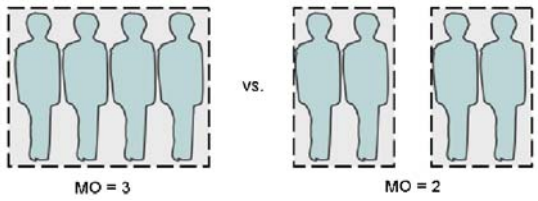

Figure 3: Counting **MO** errors. The scene on the right is "more correct" than the left.

- **Measure C-3: (MT)** - Multiple Trackers. Two or more estimates are associated with the same ground truth. A **MT** error is assigned for each excess estimate.
- **Measure C-4: (MO)** - Multiple Objects. Two or more ground truth objects are associated with the same estimate. A **MO** error is assigned for each excess ground truth.

Cases can exist where **MO** and **MT** are combined, as in the right side of Figure 2. Here, the upper $\mathcal{E}$ is tracking both $\mathcal{GT}$s, and the right $\mathcal{GT}$ is tracked by two $\mathcal{E}$s. This situation would result in two errors, one **MO** and one **MT**.

MO and **MT** errors both give an error for each excess object/estimate. This most closely matches the human intuition for correct configuration, though other methods could be considered. In Figure 3 two scenarios are presented, each with errors in configuration. On the left, one $\mathcal{E}$ covers four $\mathcal{GT}$s. On the right, two $\mathcal{E}$s cover two $\mathcal{GT}$s each. To a human observer, the scene on the right is "more correct" than the scene on the left. Using our convention, the scene on the left produces 3 **MO** errors, and the scene on the right produces 2 **MO** errors, matching human intuition. Another option could be to assign just one error, irrespective of how many objects were covered. This would produce the undesirable result of 1 **MO** error on the left and 2 **MO** errors on right. A third option could be to assign an error for each object covered by the estimate. This option also yields undesirable results, where both scenes produce 4 **MO** errors.

In [3], **FP** and **FN** errors are defined in such a way that they can not be computed each frame, and are based on overlap, not a combination of ν and ρ, such as $F_{i,j}$.

3.2 Configuration Maps

Now that the configuration errors are defined, we must establish an effective, automatic procedure for counting them. This is done by setting associations between $\mathcal{GT}$s and $\mathcal{E}$s, called *configuration maps*. Locating configuration errors then becomes a simple matter of inspecting the configuration maps.

A configuration map can be thought of as a table whose entries indicate tracking associations. Configuration maps can be constructed from the perspective of the $\mathcal{GT}$s or the $\mathcal{E}$s. Each column of a configuration map w.r.t. $\mathcal{E}$ indicates which $\mathcal{GT}$s passed the coverage test (as seen in Table 1 for the example of Figure 2). In the other direction, each column of a configuration map w.r.t. $\mathcal{GT}$ indicates $\mathcal{E}$s which passed the coverage test (Table 2). The procedure for constructing the configuration maps is outlined in Figure 4.

	estimate $\mathcal{E}$				
	1	2	3	4	5
$\mathcal{GT}$	a	b	b	-	d,e
				↑	↑
				FP	**MO**

Table 1: Configuration map w.r.t. $\mathcal{E}$ for Figure 2.

	ground truth $\mathcal{GT}$				
	a	b	c	d	e
$\mathcal{E}$	1	2,3	-	5	5
		↑	↑		
		MT	**FN**		

Table 2: Configuration map w.r.t. $\mathcal{GT}$ for Figure 2.

The four configuration error types can be inferred from inspection of the configuration maps. For configuration maps w.r.t. $\mathcal{E}$, a *false positive* (**FP**) is identified by a column with blank entry (the $\mathcal{E}$ is not tracking any object; estimate 4 in the example). A *multiple object* (**MO**) error is identified by a column with multiple entries (indicating an estimate is tracking multiple objects; estimate 5 covers both objects d and e in the example). The map w.r.t. $\mathcal{GT}$ can be used to detect **FP** and **MT** errors. A *false negative* (**FN**) is identified by a column with a blank entry (the $\mathcal{GT}$ is not being tracked; c in the example). A *multiple tracker* (**MT**) error is indicated by a column with multiple entries (the $\mathcal{GT}$ is being tracked by multiple $\mathcal{E}$s; in the example, object b is tracked by estimates 2 and 3).

`Configuration Map` w.r.t. $\mathcal{E}$

- `for` each estimate, $\mathcal{E}_i$
  - `calculate` the F-measure with each ground truth object $F_{i,j} = \frac{2\nu_{i,j}\rho_{i,j}}{\nu_{i,j}+\rho_{i,j}}$.
  - `map` $\mathcal{GT}_j \rightarrow \mathcal{E}_i$ if $F_{i,j} > t_c$.
- `end`

Figure 4: Procedure for establishing configuration mappings. The Configuration Map w.r.t. $\mathcal{GT}$ is built in a similar manner for each $\mathcal{GT}$.

3.3 Configuration Distance

We previously established that a correct tracking configuration has one $\mathcal{E}$ to every $\mathcal{GT}$. To assess the overall configuration, one can measure the difference between the number of $\mathcal{GT}$s and the number of $\mathcal{E}$s.

- **Measure C-5: (CD)** - Configuration Distance. The difference between the number of $\mathcal{E}$s and $\mathcal{GT}$s normal-

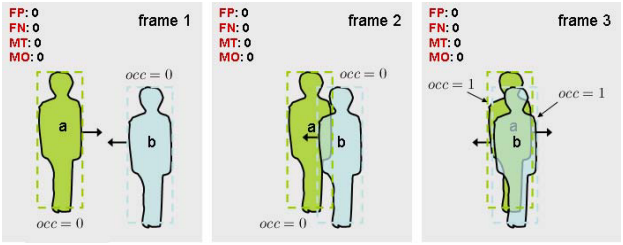

Figure 5: *Occlusion.* Arrows indicate the direction of motion of the objects. See text for details.

ized by the instantaneous number of $\mathcal{GT}$s in a given frame. Specifically,

$$\mathbf{CD} = \frac{N_{\mathcal{E}}^t - N_{\mathcal{GT}}^t}{max(N_{\mathcal{GT}}^t, 1)} \quad (3)$$

where $N_{\mathcal{E}}^t$ is the number of $\mathcal{E}$s in frame t, and $N_{\mathcal{GT}}^t$ is the number of $\mathcal{GT}$s. The **CD** should be zero when the number of $\mathcal{GT}$s and $\mathcal{E}$s is the same. It is negative when $N_{\mathcal{GT}}^t > N_{\mathcal{E}}^t$ and positive when $N_{\mathcal{GT}}^t < N_{\mathcal{E}}^t$. An absolute value is not applied to the numerator here, but later when averaging the **CD** over time (Section 3.5), so that the negative and positive values of **CD** at each frame can indicate the direction of failure. The denominator is $max(N_{\mathcal{GT}}^t, 1)$ to prevent infinite values when $N_{\mathcal{GT}}^t = 0$. In some cases, the **CD** can be misleading. In a scene with $N_{\mathcal{GT}}^t = 4$, a system might track three $\mathcal{GT}$s correctly, miss one (**FN**), and produce a **FP** (making $N_{\mathcal{E}}^t = 4$). This example yields a $\mathbf{CD} = 0$, indicating perfect configuration, when in fact two errors have been made. For this reason it is important to report the **CD** in conjunction with the configuration errors.

3.4 Occlusion

Occlusion is a special case that can cause spurious errors to appear when evaluating the configuration. When two $\mathcal{GT}$s overlap, $\mathcal{E}$s which are correctly configured will also overlap. This can generate **MT** and **MO** errors, though the configuration is correct. To handle this situation, an *occlusion flag*, *occ*, is defined so that when two $\mathcal{GT}$s (j and k) overlap above a threshold t_O, **MT** and **MO** which involve $\mathcal{GT}_j$ or $\mathcal{GT}_k$ will not be generated. The *occ* is defined directionally per object instead of pairwise to handle the case of multiple occlusions.

$$occ_j = \begin{cases} 1, & \exists\, \mathcal{GT}_k \text{ s.t.} |\mathcal{GT}_j \cap \mathcal{GT}_k| > t_O \\ 0, & otherwise \end{cases} \quad (4)$$

An example is shown in Figure 5 ($t_C = 50\%$, $t_O = 80\%$) where objects a and b approach each other in the first frame, partially occlude in the second (but neither t_C or t_O is exceeded), and nearly fully occlude in the third. The *occ* flag prevents errors from being produced in the third frame.

3.5 Procedure and Example

The procedure for evaluating the configuration of a tracking sequence is presented using the example in Figure 6,

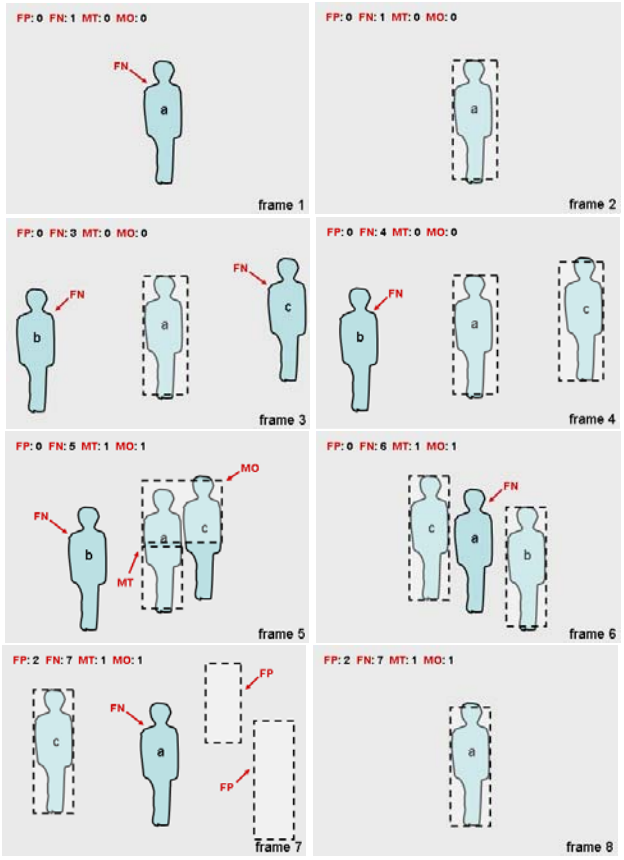

Figure 6: Configuration errors in a hypothetical tracking scenario. Errors in upper-left indicate accumulation of errors over the sequence. (*Frame 1*) a stands alone untracked, generating a **FN**. (*Frame 2*) An $\mathcal{E}$ begins to track a. (*Frame 3*) b and c enter untracked from either side, each generating a **FN**. (*Frame 4*) An $\mathcal{E}$ begins to track c; b remains untracked. (*Frame 5*) Upper $\mathcal{E}$ tracks a and c causing a **MO**, a is tracked by 2 $\mathcal{E}$s causing a **MT**. (*Frame 6*) c and b are tracked. a produces a **FN**. (*Frame 7*) b leaves the scene; two lost $\mathcal{E}$s cause **FP**s on the right; a remains untracked. (*Frame 8*) c leaves the scene; a is tracked correctly.

where three objects (a,b,c) are tracked over eight frames. Configuration errors are indicated in red. The procedure for configuration evaluation itself is outlined in Figure 7.

In steps 3 and 4 of Figure 7, the basic configuration measures are normalized by $max(N_{\mathcal{GT}}^t, 1)$ and averaged over the length of the sequence (except for **CD**). As with **CD** in Section 3.3, the denominator prevents infinite values when no $\mathcal{GT}s$ are present (whereas [3] can not evaluate frames where $N_{\mathcal{GT}}^t = 0$). These normalized measures are reported to facilitate the comparison of performance over datasets with varying length and numbers of objects. Results from the example are simple to compute, shown in Table 3.

In Figure 8, the **CD** and sum of errors normalized each frame by $max(N_{\mathcal{GT}}, 1)$ is plotted against time. Graphs like this can be useful to showcase the performance of a tracking method on a short sequence, but not for long sequences or batches of experiments.

Procedure for evaluating configuration.

1. Construct configuration maps for each time step.
2. Using configuration maps, determine **FP**, **FN**, **MT**, and **MO** errors for each frame as described in Section 3.2.
3. Report $\overline{\mathbf{FP}}$, $\overline{\mathbf{FN}}$, $\overline{\mathbf{MT}}$, and $\overline{\mathbf{MO}}$: the accumulation of each error type normalized by $max(N^t_{\mathcal{GT}}, 1)$ each frame, averaged over the total number of frames, n.

$$\overline{\mathbf{FP}} = \frac{1}{n}\sum_{t=0}^{n}\frac{\mathbf{FP}^t}{max(N^t_{\mathcal{GT}},1)},\ \overline{\mathbf{FN}} = \frac{1}{n}\sum_{t=0}^{n}\frac{\mathbf{FN}^t}{max(N^t_{\mathcal{GT}},1)},$$
$$\overline{\mathbf{MT}} = \frac{1}{n}\sum_{t=0}^{n}\frac{\mathbf{MT}^t}{max(N^t_{\mathcal{GT}},1)},\ \overline{\mathbf{MO}} = \frac{1}{n}\sum_{t=0}^{n}\frac{\mathbf{MO}^t}{max(N^t_{\mathcal{GT}},1)} \quad (5)$$

4. Calculate **CD** for each frame and report $\overline{\mathbf{CD}}$.

$$\overline{\mathbf{CD}} = \frac{1}{n}\sum_{t=1}^{n}|\mathbf{CD}| \quad (6)$$

Figure 7: Procedure for evaluating configuration.

FP	FN	MT	MO	CD
2	7	1	1	-5

$\overline{\mathbf{FP}}$	$\overline{\mathbf{FN}}$	$\overline{\mathbf{MT}}$	$\overline{\mathbf{MO}}$	$\overline{\mathbf{CD}}$
.13	.40	.04	.04	0.35

Table 3: Results for example of Figure 6. *(top)* Accumulation of errors over 8 frames. *(bottom)* Time-averaged errors from step 3 of Figure 7.

3.6 Track State

The *track state*, τ, defined for each $\mathcal{GT}$, is a simple binary variable indicating if a $\mathcal{GT}$ is being tracked or not. While not a measure in itself, it will be useful for (1) setting rules when to evaluate task-specific performance measures in Section 5 (e.g., rule: only compare the velocity of $\mathcal{E}_i$ to $\mathcal{GT}_j$ when $\tau_j = 1$), and (2) to visualize performance. Track state is related to the configuration in that it provides a history of *if* and *when* $\mathcal{GT}$s are tracked. Track state is defined as

$$\tau_j = \begin{cases} 1, & \exists\ \mathcal{E}_i \text{ s.t. } F_{i,j} > t_C \\ 0, & otherwise \end{cases} \quad (7)$$

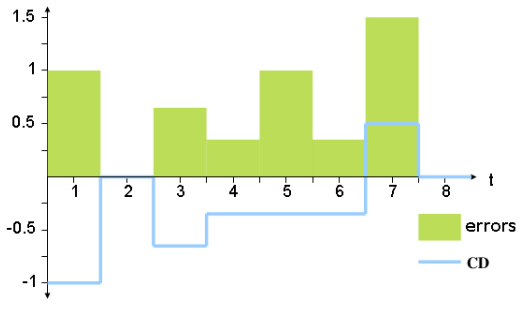

Figure 8: Configuration errors and **CD** reported for each frame for example of Figure 6.

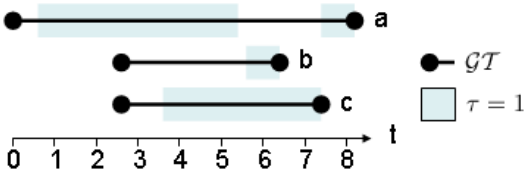

Figure 9: Tracking results can be visualized by plotting τ_j with the presence of $\mathcal{GT}$s.

In some cases, it may be useful to visualize the temporal evolution of τ for several objects. Plotting τ and the presence of $\mathcal{GT}$s allows one to visualize scene entrances/exits, and temporary tracking losses. An example can be seen in Figure 9.

4 Identification

In object tracking, identification implies the persistent tracking of an object by a particular estimate over time. This differs from configuration, which is concerned only with the spatial relations between $\mathcal{E}$s and $\mathcal{GT}$s. Identification is concerned with both temporal and spatial relations. It is important to note the difference between this type of identification and identification in the recognition sense (i.e. "estimate 1 is Bill Clinton"). In this section, we will present four identification measures (labeled I-1 to I-4) and a procedure for evaluating how well a tracker identified objects in the scene.

4.1 Identification Errors

Examining the modes of failure of a path matching system whose sole task is to correctly identify $\mathcal{E}$s and $\mathcal{GT}$s in a sequence can assist in understanding the types of identification errors. This path matcher must examine a set of $\mathcal{E}$ and $\mathcal{GT}$ trajectories to determine which $\mathcal{E}$ trajectories identify each $\mathcal{GT}$ trajectory, and vice versa. Ideally, if the configuration and identification are correct throughout, each $\mathcal{GT}_j$ is tracked by exactly one particular $\mathcal{E}_i$ over its entire lifetime. In this case, $\mathcal{GT}_j$ is said to be *identified* by $\mathcal{E}_i$ and that $\mathcal{E}_i$ is said to be *identifying* $\mathcal{GT}_j$. An important intrinsic property of identity is that each $\mathcal{E}$ can identify at most one $\mathcal{GT}$, and that each $\mathcal{GT}$ can be identified by at most one $\mathcal{E}$, though they do not necessarily have to match.

There are two types of identification errors that can be produced by the path matcher. First, it is possible that at some point, $\mathcal{GT}_j$ is tracked by some other estimate, $\mathcal{E}_k$. An example of this is shown in Figure 10 where object a is identified by estimate 1 (blue), but later tracked by estimate 2 (red). The other error type occurs when $\mathcal{E}_i$ tracks $\mathcal{GT}_j$ for a while, but then "swaps" ground truths and begins tracking $\mathcal{GT}_l$. Swapping is a common problem in tracking, and can be seen in Figure 10 where estimate 3 (green) swaps from object b to c. Thus, we can define two types of identification errors:

- **Measure I-1: (FIT)** - Falsely Identified Tracker. An $\mathcal{E}_i$ segment which passes the coverage test for $\mathcal{GT}_j$ but

is not the *identifying* $\mathcal{E}$.

- **Measure I-2: (FIO)** - Falsely Identified Object. A $\mathcal{GT}_j$ segment which passes the coverage test for $\mathcal{E}_i$ but is not the *identified* $\mathcal{GT}$.

For an $\mathcal{E}$ to identify a $\mathcal{GT}$ or for a $\mathcal{GT}$ to be identified by an $\mathcal{E}$, they must be associated in the *identity maps*, which are defined in the next section. A segment is defined as a consecutive set of similarly labeled frames (e.g. the frames where green tracks b in Figure 10 is a segment). In [3], errors similar to **FIT** and **FIO** are defined which differ in the mapping method (an ad-hoc temporal threshold is used).

4.2 Identity Maps

In the example of Figure 10, it is fairly obvious which $\mathcal{E}$s truly identify each $\mathcal{GT}$, and vice versa. However, this is not always the case. Sometimes, in the face of poor tracking, it can be difficult even for a human to determine proper associations. Cases can arise where identification is not reciprocated, for instance $\mathcal{GT}_j$ might be best identified by $\mathcal{E}_i$, but $\mathcal{E}_i$ is better described as identifying $\mathcal{GT}_k$.

Finding a good method to associate identities is critical. Several methods could be considered, such as an instant association (as soon as an $\mathcal{E}$ appears, it identifies the first $\mathcal{GT}$ for which it passes the coverage test), delayed association (when an $\mathcal{E}$ appears, it identifies the first $\mathcal{GT}$ for which it passes the *coverage test* after an arbitrary time delay), and exit association (each $\mathcal{E}$ identifies the last $\mathcal{GT}$ for which it passed the *coverage test*). However, each of these rules introduce undesirable flaws and inconsistencies. To avoid such problems, we form identification associations on a "majority rule" basis, where an $\mathcal{E}$ identifies the $\mathcal{GT}$ it spends the most time tracking, and a $\mathcal{GT}$ is identified by the $\mathcal{E}$ which tracks it the largest amount of time (tracking implies passing the coverage test).

Identity maps, in a similar manner to configuration maps, can be thought of as a table describing identity associations between $\mathcal{E}$s and $\mathcal{GT}$s. As with configuration maps, there exist two directions from which maps can be created (w.r.t. $\mathcal{GT}$ and w.r.t. $\mathcal{E}$). The map w.r.t. $\mathcal{E}$ is formed by first collecting all the $\mathcal{GT}$s which pass the coverage test for $\mathcal{E}_i$ into n_{ij},

$$n_{ij} = \sum_t \sum_j \mathbb{1}(F^t_{i,j} > t_c) \tag{8}$$

where $\mathbb{1}$ is the indicator function. A winning $\mathcal{GT}$, $\hat{j}_i$, is determined from n_{ij} for $\mathcal{E}_i$, and mappings are established as described in Figure 11. A similar procedure is done for the map w.r.t. $\mathcal{GT}$.

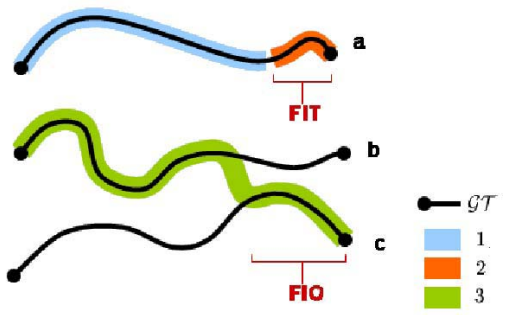

Figure 10: Types of identification errors. Three objects are tracked by three estimates. A falsely identified tracker error (**FIT**) occurs when object a is tracked by a second estimate, $\mathcal{E}_2$. A falsely identified object error (**FIO**) occurs when an estimate swaps ground truths ($\mathcal{E}_3$ swaps from $\mathcal{GT}_b$ to $\mathcal{GT}_c$).

`Identification Map` w.r.t. estimate $\mathcal{E}$

- `for` each estimate, $\mathcal{E}_i$
  - `collect` $\mathcal{GT}$s which pass the coverage test into a function $n_{ij} = \sum_t \sum_j \mathbb{1}(F^t_{i,j} > t_C])$
  - `determine` the majority rules winning estimate $\hat{j}_i = \arg\max_j n_{ij}$
  - `map` $\mathcal{E}_i \rightarrow \mathcal{GT}_{\hat{j}_i}$
- `end`

`Identification Map` w.r.t. ground truth $\mathcal{GT}$

- `for` each ground truth object, $\mathcal{GT}_j$
  - `collect` $\mathcal{E}$s which pass the coverage test into a function $n_{ji} = \sum_t \sum_i \mathbb{1}(F^t_{i,j} > t_C])$
  - `determine` the majority rules winning estimate $\hat{i}_j = \arg\max_i n_{ji}$
  - `map` $\mathcal{GT}_j \rightarrow \mathcal{E}_{\hat{i}_j}$
- `end`

Figure 11: Procedure for establishing identification mappings.

4.3 Purity

Purity is a measure that can be used to evaluate the degree of consistency to which an $\mathcal{E}$ correctly identifies a $\mathcal{GT}$, and vice versa. We define purity for object identification similarly to the purity concept which is used to evaluate speaker clustering [8]. Purity can be evaluated w.r.t. $\mathcal{E}$ and w.r.t. $\mathcal{GT}$.

- **Measure I-3: (TP)** Tracker Purity. If the identity map w.r.t. $\mathcal{E}$ indicates $\mathcal{E}_i$ identifies $\mathcal{GT}_j$, the ratio of frames that $\mathcal{E}_i$ correctly identifies $\mathcal{GT}_j$ ($n_{i\hat{j}_i}$) to the total number of frames $\mathcal{E}_i$ exists (n_i) .

$$\mathbf{TP} = \frac{n_{i\hat{j}_i}}{n_i} \tag{9}$$

- **Measure I-4: (OP)** - Object Purity. If the identity map w.r.t. $\mathcal{GT}$ indicates $\mathcal{GT}_j$ is identified by $\mathcal{E}_i$, the ratio of frames that $\mathcal{GT}_j$ is correctly identified by $\mathcal{E}_i$ ($n_{j\hat{i}_j}$) to the total number of frames $\mathcal{GT}_j$ exists (n_j) .

$$\mathbf{OP} = \frac{n_{j\hat{i}_j}}{n_j} \tag{10}$$

4.4 Procedure and Example

The procedure for identification evaluation is shown in Figure 12. An example is shown in Figure 13 where three $\mathcal{GT}$s are tracked and identified by four $\mathcal{E}$s.

`Procedure for evaluating identification.`

1. Construct identity maps for entire sequence, as outlined in Section 4.2.
2. Assign **FIT** and **FIO** errors for each frame by checking if $\mathcal{E}$ and $\mathcal{GT}$ segments which pass the coverage test match $\mathcal{E}$s and $\mathcal{GT}$s indicated by the identity maps. Label segments as **FIT**, **FIO**, or correct.
3. Report $\overline{\mathbf{FIT}}$ and $\overline{\mathbf{FIO}}$, the accumulation of each error type normalized by the frame count n and $max(N^t_{\mathcal{GT}}, 1)$.

$$\overline{\mathbf{FIT}} = \frac{1}{n}\sum_{t=0}^{n}\frac{\mathbf{FIT}_t}{max(N^t_{\mathcal{GT}},1)}, \quad \overline{\mathbf{FIO}} = \frac{1}{n}\sum_{t=0}^{n}\frac{\mathbf{FIO}_t}{max(N^t_{\mathcal{GT}},1)} \tag{11}$$

4. Calculate **TP** for each $\mathcal{E}$ and **OP** for each $\mathcal{GT}$. Report *average tracker purity* $\overline{\mathbf{TP}}$ and *average object purity* $\overline{\mathbf{OP}}$.

$$\overline{\mathbf{TP}} = \frac{1}{N_{\mathcal{E}}}\sum_{i=0}^{N_{\mathcal{E}}}\mathbf{TP}, \quad \overline{\mathbf{OP}} = \frac{1}{N_{\mathcal{GT}}}\sum_{j=0}^{N_{\mathcal{GT}}}\mathbf{OP} \tag{12}$$

Figure 12: Procedure for evaluating identification.

In step 3 the errors are normalized in a similar manner to the configuration errors in Section 3.5 to facilitate comparisons over data sets of differing length and content. In step 4, the tracker purity is normalized by the total number of $\mathcal{E}$s appearing in the sequence ($N_{\mathcal{E}}$), and the object purity is normalized by the total number of objects in the sequence ($N_{\mathcal{GT}}$).

To assist in the evaluation, an *identification graph* can be constructed, which visually describes tracking associations between $\mathcal{E}$s and $\mathcal{GT}$s (see Figure 14). These associations can be determined by performing coverage tests every frame on each $\mathcal{E}$ and $\mathcal{GT}$. The presence of ground truth objects are represented by solid lines, and associated $\mathcal{E}$s are shown as surrounding colored bars. Looking at the identification graph, it is obvious how identity maps are generated. The map w.r.t. $\mathcal{GT}$ is created by determining for each $\mathcal{GT}$, which $\mathcal{E}$ tracks it the longest. Inspecting Figure 14, object a is clearly tracked by 1 the longest, b tracked by 2, and c by 3. Thus, the *identity map* w.r.t. $\mathcal{GT}$ is:

	ground truth $\mathcal{GT}$		
	a	b	c
$\mathcal{E}$	1	2	3

In the other direction, the map w.r.t. $\mathcal{E}$ is created by determining for each $\mathcal{E}$, what $\mathcal{GT}$ it tracked the most. In Figure 14, 1 tracked a the longest, 2 tracked b, 3 tracked c, and 4 also tracked b. The *identity map* w.r.t. $\mathcal{E}$ is:

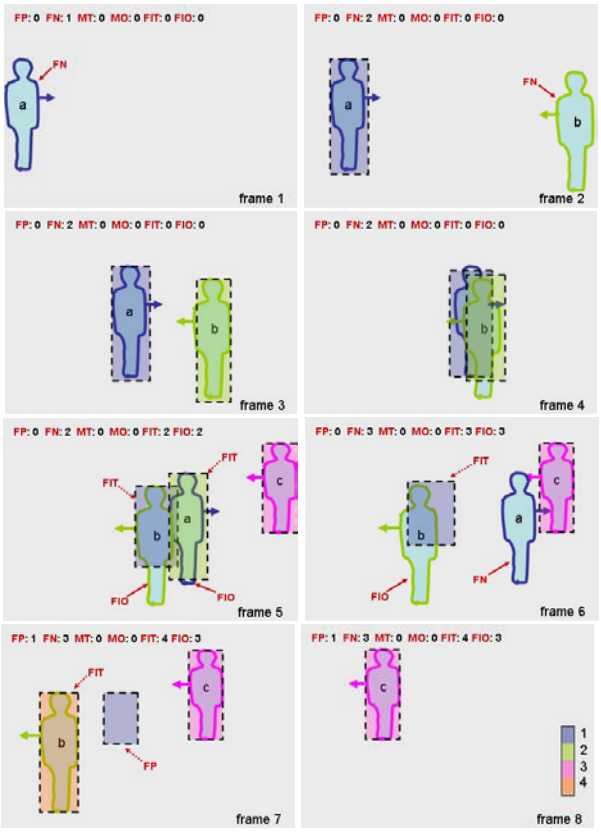

Figure 13: *Identification*. A tracking scenario in which four $\mathcal{E}$s (1,2,3,4) track three $\mathcal{GT}$s (a, b, c). (*Frame 1*) a enters from the left, untracked. (*Frame 2*) a is correctly tracked and identified by 1, b enters from the right. (*Frame 3*) a & b approach each other, 2 tracks and correctly identifies b. (*Frame 4*) a & b pass each other. (*Frame 5*) After passing, 1 swaps from a to b, and 2 swaps from b to a, generating **FIT** & **FIO** errors. c enters from the right correctly tracked and identified by 3. (*Frame 6*) 2 dies, a approaches c untracked, 1 continues to misidentify b. (*Frame 7*) 1 becomes lost and generates a **FP**. 4 appears and tracks b, generating a **FIT**. (*Frame 8*) b exits the scene, leaving c correctly tracked and identified.

	estimate $\mathcal{E}$			
	1	2	3	4
$\mathcal{GT}$	a	b	c	b

The next step is to locate identification errors and label the segments as **FIT**, **FIO**, or correct. Looking at the identification graph can assist in understanding error location and segment labeling. First, correctly identified $\mathcal{GT}$ segments are found by a table lookup on the map w.r.t. $\mathcal{GT}$; all segments where $\mathcal{GT}_j$ is tracked by $\mathcal{E}_{\hat{i}_j}$ are labeled as correct (e.g. where *blue* tracks a in the lower left of Figure 14). Segments where $\mathcal{GT}_j$ is tracked by other $\mathcal{E}$s are labeled as **FIT**s (e.g. after a and b cross in Frame 5, a is misidentified by 2). **FIO** errors are labeled in a similar manner. A table lookup on the map w.r.t. $\mathcal{E}$ is done to find $\mathcal{GT}_{\hat{j}_i}$ for each $\mathcal{E}_i$. $\mathcal{GT}$ segments matching $\hat{j}_i$ are labeled as correct (e.g. 1 correctly identifies a in the lower right of Figure 14), others are labeled as **FIO** errors (e.g. 1 misidentifies b after swapping in Frame 5). Finally, results from steps 3 & 4 of the procedure are computed and reported, as shown in Table 4.

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)
1063-6919/05 $20.00 © 2005 **IEEE**

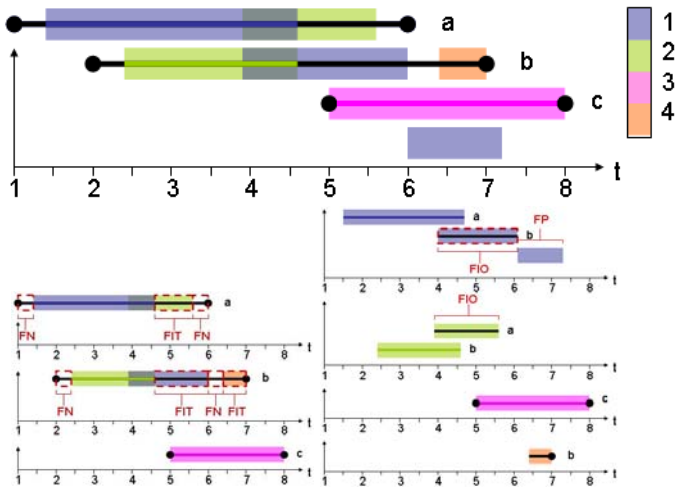

Figure 14: (top) *Identification graph* for example in Figure 13. The presence of $\mathcal{GT}$ objects is denoted by solid black lines. Associated $\mathcal{E}$s are represented by colored bars. (lower left) Errors viewed w.r.t. $\mathcal{GT}$. (lower right) Errors viewed w.r.t. $\mathcal{E}$.

FP	FN	MT	MO	CD	FIT	FIO
1	3	0	0	-2	4	3

$\overline{\text{FP}}$	$\overline{\text{FN}}$	$\overline{\text{MT}}$	$\overline{\text{MO}}$	$\overline{\text{CD}}$	$\overline{\text{FIT}}$	$\overline{\text{FIO}}$	$\overline{\text{TP}}$	$\overline{\text{TO}}$
.06	.23	0	0	0.29	.19	.13	.79	.61

Table 4: Results for example of Figure 6. *(top* Accumulation of errors over 8 frames. *(bottom)* Errors from step 3 of Figure 12.

4.5 Identity State

The *identity state*, id, is a binary variable that operates in a similar manner to the track state ($id = 1$ for correct identification, 0 otherwise). The identity state is defined for each $\mathcal{GT}$, but requires correct identification in both directions ($\mathcal{E}_i = \mathcal{E}_{\hat{i}_j}$ w.r.t. $\mathcal{GT}$, and $\mathcal{GT}_j = \mathcal{GT}_{\hat{j}_i}$ w.r.t. $\mathcal{E}$). Like τ, the identity state is useful for setting rules determining when to evaluate task-specific measures.

$$id_j^t = \begin{cases} 1, & \exists\ \mathcal{E}_i \text{ s.t. } j = \hat{j}_i \text{ and } i = \hat{i}_j \\ 0, & otherwise \end{cases} \quad (13)$$

5 Task-Specific Measures

Many performance measures are often task-dependent. Some task-dependent measures may have a physical meaning, for example: object velocity, head angle, etc. Other task-specific measures might include frame-based labeling tasks. It is not within the scope of this document to anticipate the multitude of possible tracking tasks and define appropriate measures for each; this framework allows users to define these measures themselves.

The track state τ_j^t and identity state id_j^t can be used to assist in the definition of task-specific measures and to automate their calculation. They can be used as a switch to decide if task-specific performance calculations should be performed on a given estimate-object pair (i, j) at time t. For instance, when trying to measure the difference between the estimated $v_{\mathcal{E}}$ and ground truth velocity $v_{\mathcal{GT}}$ of an object, it makes sense to make calculations only for frames where the $\mathcal{GT}$ is being tracked ($\tau_j^t = 1$). When $\tau_j^t = 0$, no $\mathcal{E}$ is tracking the $\mathcal{GT}$, and hence no $v_{\mathcal{E}}$ is available. The *identity state* id_j^t can be used in a similar manner to compute measures when a $\mathcal{GT}$ is correctly identified. In order to run batches of experiments on the same dataset, τ_j^t and id_j^t are expressed with respect to the $\mathcal{GT}$, as the number of $\mathcal{GT}$s remains static but the $\mathcal{E}$s do not.

Finally, F_{ij} can be used to evaluate the "quality of tracking". Higher values of F_{ij} indicate better tracking. Setting the criteria that the F-measure is only reported when $\tau_j^t = 1$ ensures that only valid pairs of $\mathcal{E}$s and $\mathcal{GT}$s are evaluated in a given frame.

6 Conclusion

In this paper, we have defined a set of measures and protocols for evaluating the configuration and identification performance of multi-object tracking systems, in an effort to address the lack of uniform metrics in the tracking community. In our view, these measures and protocols are meaningful to both academic and real-life evaluation, and are defined within a broader framework which is designed to evaluate all important aspects of tracking performance. Evaluations of several tracking methods using these measures and related software can be found at http://www.idiap.ch/∼smith.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto *Modern Information Retrieval*, ACM Press, 1999.

[2] J. Black et al., "A Novel Method for Video Tracking Performance Evaluation," in *Proc. IEEE PETS Workshop*, Oct. 2003.

[3] L.M. Brown et al. "Performance Evaluation of Surveillance Systems Under Varying Conditions," in *Proc. IEEE PETS Workshop*, Jan. 2005.

[4] R. Collins, X Zhou, S. Teh, "An Open Source Tracking Testbed and Evaluation Web Site," in *Proc. IEEE PETS Workshop*, Jan 2005.

[5] R. B. Fisher, "The PETS04 Surveillance Ground-Truth Data Sets", in *Proc. IEEE PETS Workshop*, May 2004.

[6] J. Nascimento and J. S. Marques, "New Performance Evaluation Metrics for Object Detection Algorithms," in *Proc. IEEE PETS Workshop*, May 2004.

[7] J. Nurminen, "Using Software Complexity Measures to Analyze Algorithms," *Computers and Operation Research* vol. 30, i. 8, pp 1121-1134, July 2003.

[8] A. Solomonoff, et al., "Clustering Speakers by their Voices" in *Proc. IEEE ICASSP*, 1998.