CHARACTERISATION OF TRACKING PERFORMANCE

Andrea Cavallaro[1] and Francesco Ziliani[2]

[1]Multimedia and Vision Lab, Queen Mary, University of London
Mile End Road, London E1 4NS (United Kingdom) - Email: andrea.cavallaro@elec.qmul.ac.uk
[2]VisioWave
CH - 1024 Ecublens (Switzerland) - Email: francesco.ziliani@visiowave.com

ABSTRACT

The accuracy of tracking algorithms is highly data dependent. However, tracking algorithms are often demonstrated using a small data set that cannot provide significant statistical performance measures. In addition to this, it is common to evaluate and compare results by visual analysis only. In this paper, we propose a benchmarking protocol to evaluate and compare object tracking algorithms. In particular, we propose to separate the evaluation problem in two parts, namely algorithmic evaluation and application-dependent evaluation. Furthermore, we provide a set of scores that allow one to rate and to compare different solutions. Given the complexity of the task, the goal is not to derive a unique measure of performance, but a combination of scores that reflect the behavior of the specific algorithm. This benchmarking protocol enables the comparison of different algorithms, the understanding of their limits, and the monitoring of the technological progress in the field.

1. INTRODUCTION

An increasing number of algorithms for object tracking have been proposed in the last decade. However, most of the time new algorithms or improvements of existing ones are not tested and demonstrated using a commonly accepted evaluation protocol. Evaluating tracking algorithms is an important stage for validating incremental modification to algorithms, to compare performance of alternative approaches and to develop new approaches. An evaluation protocol should be composed of a data set and one or more evaluation metrics. A data set should be composed of test sequences and, preferably, their associated ground-truth data. Important steps in the direction of having an appropriate data set have been done with the creation of common test sequences (e.g., PETS[1], FGNET[2]). These data sets aim at covering specific applications, but do not address the problem of providing a sufficient number of statistically relevant sequences. Large corpora of test sequences should be used in order to provide conclusive validation of algorithmic robustness and flexibility. However, the definition of the type of sequences that should be included in the data set is a difficult task. Moreover, due to privacy issues, distributing data sets to the research community is problematic when people or their belongings, such as cars, are represented in the sequences.

Once the corpus has been defined, ground-truth data need to be provided to enable performance evaluation and comparisons. A number of evaluation metrics have been proposed to measure the deviation of automatically generated results and the corresponding ground-truth data, but there is not a commonly recognized measure or set of measures [1-6].

In this paper, we propose a list of benchmarking criteria to characterize tracking performance and to enable the comparison of different algorithms. The idea is to produce a general evaluation of an algorithm (tested as standalone algorithm) as well as an evaluation of the performance of the same algorithm in the context of a specific application. The rationale behind this choice is that algorithms need to be tested independently from the application to ensure flexibility and to avoid ad-hoc solutions that would work in a specific application context only. This algorithmic testing is then complemented by an evaluation in real scenarios.

The paper is organized as follows. In Section 2, definitions and notations that are used in the following section will be introduced. Section 3 describes the metric considered in the algorithmic evaluation. Section 4 introduces the evaluation in the context of a specific scenario, namely the monitoring of a subway station. Section 5 provides a discussion on the major issues in performance comparison. Finally, Section 6 concludes the paper.

2. PROBLEM STATEMENT

The output of a tracking algorithm is the estimated location in the image plane of an object or of a part of an object over time. The object or a part of it (e.g., a face or a hand) to be tracked is referred to as target. The location of the target is represented with a 2D position (coordinates in the image plane) or with a 2D shape representation. The shape representation of the target can take the form of a bitmap [7], a polygonal approximation, a bounding box or an ellipse (Figure 1). When the output of a tracker is the predicted shape of the target (or its approximation), then the 2D position can be derived in a number of ways, for instance by computing the center of mass of the shape, the intersection of the diagonals of the bounding box, or the centre of the ellipse.

[1] www.visualsurveillance.org

[2] www.fg-net.org

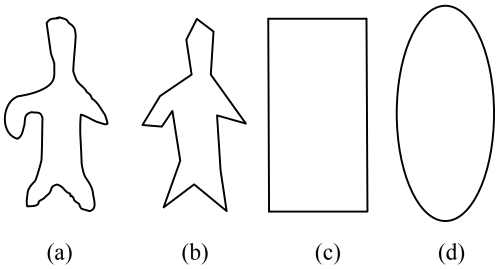


Figure 1: Shape representation of a target. (a) bitmap; (b) poligonal approximation; (c) bounding box; (d) ellipse.

A trajectory is the ordered temporal collection of the estimated target locations. A general representation of a trajectory T is T={(x_i,y_i,S_i): n_1<i<n_2}, where (x_i,y_i) is the estimated position of the target in the image plane and S_i the estimated shape. In the case of the elliptical shape approximation Si = (a_i, b_i, θ_i), where a_i and b_i are the two axes of the ellipse and θ_i is its orientation in the image plane with respect to the x axis. The time interval τ = n_2-n_1 is the life-span of the trajectory.

In order to assess the goodness of a predicted trajectory, it is desirable to compare it with a reference trajectory (ground-truth). A ground-truth is the ideal representation of a target over time and is generally defined manually. Although there is a certain degree of subjectivity in the generation of the ground-truth - different persons could draw different ground-truths and the same person could draw different ground-truths at different time instants – we will consider the ground truth as the *ideal* representation of a target over time.

3. ALGORITHMIC EVALUATION

Algorithmic evaluation is the application-free phase of performance characterization. Application-free evaluation aims at testing a number of quality factors that measure the goodness of the results produced by a certain tracking algorithm. The objective is to evaluate the performance of a given algorithm when changing parameters and the input data. This evaluation uses a set of metrics. The metrics quantify the accuracy, the stability, the robustness and the complexity of the algorithm. Accuracy, stability and robustness are computed based on a ground-truth. Note that the ground-truth used in the algorithmic evaluation is independent from the application. In Section 4 we will define an application-dependent ground-truth.

The *accuracy* is measured in terms of position error and size error. The *position error*, p_e, is the deviation of the predicted trajectory from the ground-truth trajectory (Figure 2(b)). For instance, the position error is the Euclidean distance between the centre of mass of the bounding boxes of the estimated and the ground-truth target. The *size error*, s_e, is expressed in terms of false positives and false negatives (Figure 2(c)). A false positive is a pixel erroneously detected as part of the target. A false negative is a pixel erroneously *not* detected as belonging to the target [8]. The size error is normalized by the real and predicted size of the target and can be computed as $s_{e=} (f_p+f_n)/(A_g+A_t)$, where f_p and f_n are the number of false positives and false negatives, respectively; A_g and A_t are the number of pixels in the ground-truth target and in the predicted target, respectively. When the target is represented with a fixed geometrical shape, such as a rectangular or an ellipse, then the *rotation error*, $r_{e,}$ is considered as well (Figure 2(b)). The accuracy *(p_e s_e r_e)* is measured for each frame of the test sequence (the time variable is omitted for simplicity of notation). A measure of the accuracy changes over time quantifies the *stability* of the tracking algorithm.

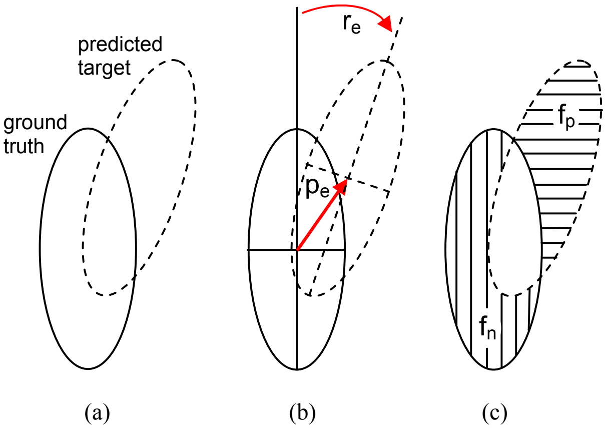


Figure 2: Representation of a target using an ellipse. (a) ground-truth and predicted target location; (b) position error (*p_e*) and rotation error (*r_e*); (c) false positives (*f_p*) and false negatives (*f_n*).

The *robustness* of the algorithm is measured by quantifying the changes in tracking results as a function of changes in the data. Examples of changes in the data are noise or induced errors in the initialization. The *robustness to noise*, r_n, is tested by adding different amount of noise to the sequence. This evaluation can be done directly with different test sequences as well as can be simulated by adding noise or changing the apparent illumination in a given test sequence. The value of r_n is the ratio between the number of initializations and the number of final predictions on target. The values of r_n are in the range [0, 1]. The higher r_i, the better the robustness. The analysis of r_n allows us to quantify the amount of noise that is accepted by the algorithm before loosing the track as well as to quantify the influence of a certain amount of noise on different trackers. The *robustness to error in initialization*, r_i, is tested by first generating a large number of initialization areas on and around the real target and then by counting the number of areas that reach the end of a predefined sequence on the target. Similarly, a perturbation during the sequence is added in order to test the recovery of the tracker. As for r_n, the values of r_i are in the range [0, 1].

Complexity is the measure of the execution time, $e_{t,}$ for a given target, or group of targets, in a given sequence. This measure is used to compare different algorithms on the same target or to quantify the cost of an improvement introduced in an algorithm. Complexity is used to determine the trade-off between computational time and accuracy.

Although it is possible to use automatic initialization, the above-mentioned properties are preferably tested when using manual *initialization* of the tracker. Manual initialization limits the accumulation of errors from different algorithms and allows one to concentrate the analysis on the tracking algorithm only. Then automatic initialization will be tested in the application-dependent part of the performance characterization, as described in the following section.

4. APPLICATION-DEPENDENT EVALUATION

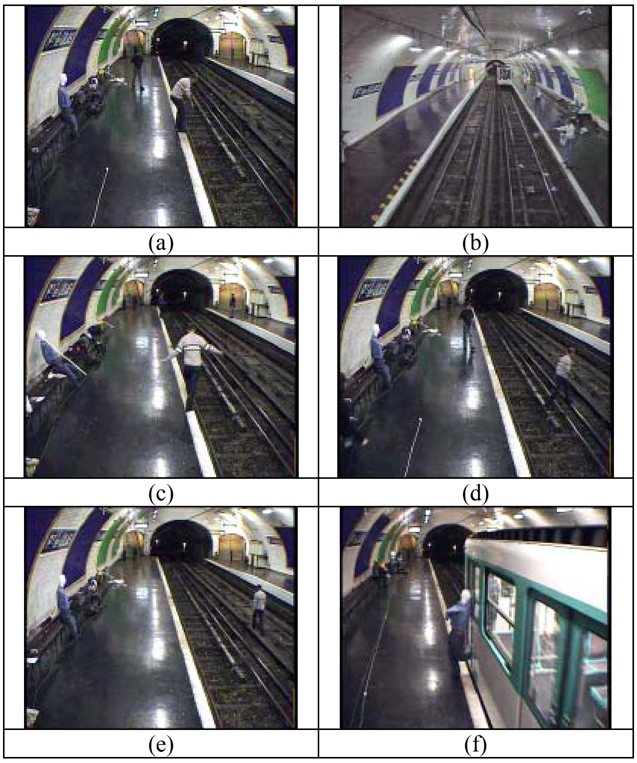
In the second part of the evaluation, a tracking algorithm is treated as a black box and its performance is evaluated in the context of a specific application. In this context, two steps are required, namely the definition of the application requirements and the definition of specific case scenarios. Based on these two steps, an application-dependent ground-truth can be defined and the relevance of each metric introduced in Section 3 can be estimated. The relevance of each metric is accounted for with a relative weight assigned to each score.

As opposed to algorithmic evaluation, the application-dependent evaluation requires that the tracking algorithm operates with automatic initialization. Moreover, the tracking algorithm will cooperate with additional analysis tools. Note that although in this paper we are focusing on the evaluation of tracking algorithms, the application-dependent evaluation is general and could also be applied to other detection methodologies.

Tracking algorithms are adopted in several surveillance scenarios to generate statistics about activities in a scene. Examples are people counting and behavior analysis. Other surveillance scenarios require automated event detection. Examples of events are accidents in a tunnel or persons crossing the rails in a subway station (Figure 3). The application-dependent evaluation in the case of event detection defines a performance score resulting from the comparison of the ground-truth data with the automatically generated data. This performance score is here referred to as *final reliability score*, s_r. The final reliability score is used to choose the best algorithm for the specific application. We report in the following the scores used in the Challenge on Real-time Event Detection Solutions (CREDS) for Enhanced Security and Safety in Public Transportation[3].

The value of s_r is computed based on the score associated to correct detections, s_c, the score associated to false positive detections, s_p, and the score associated to false negative detections, s_n. A false positive detection occurs when an event is incorrectly found where none exists in reality. A false negative detection occurs when an event is incorrectly not detected when, in fact, is present.

The score associated to correct detections, s_c, is a non-negative function of the delay/anticipation (t), the ratio between the detected event and the duration of the corresponding ground-truth event (d), and the spatial accuracy (a) between the detected event and the corresponding ground-truth event: $s_c=f(t,d,a)$. The score associated to false positive detection, s_p, and false negative detections, s_n, contribute as a penalty to the final reliability score, which can be expressed as: $s_r= f(t,d,a)- s_p- s_n$. Note that s_p and s_n take into account the definition of the specific event of interest: errors in the detection of different events of interest contribute differently in the final reliability score.



(a) (b) (c) (d) (e) (f)

Figure 3: Example of typical alarm events in video surveillance of public transportation networks (courtesy of RATP from CREDS dataset).

In the subway surveillance scenario shown in Fig. 3, the events to be detected represent normal and abnormal people behaviors, with and without the presence of a train in the station. Events of interest can be divided into three classes, namely warnings, alarms and critical alarms.

Events classified as *warnings* are suspicious or unsafe people behaviors. If detected, these events may help prevent an accident (e.g., by automatically broadcasting warning signals close to the detected event) or may provide surveillance operators with useful information to prevent or persecute crime or illicit behaviors. For this kind of events, it is acceptable that an algorithm provides non-zeros false negative detection rates and false positive detection rates.

Events classified as *alarms* correspond to events such as person trapped in the door of a moving train or people throwing objects across platforms. It is important to detect these events as soon as possible and to request an immediate visual verification from the surveillance operators. In this case, it is important to limit false negative detection rates to 0%, while false positive detection rates larger then 0% can be tolerated because a visual inspection is required before taking any actions.

Events classified as *critical alarms* correspond to events where a person is in danger: immediate and automatic action is

[3] http://www-dsp.elet.polimi.it/avss2005/

required, such as an emergency cut of the power supply on the rails or the activation of the emergency breaks of the train. These events include people crossing the rails or falling on the rails.

Each class of events may tolerate a different performance from the tracking and event detection algorithm. This observation is taken into account in the definition of the parameters that characterize the scores s_c, s_p, and s_n. Moreover, one or more events may be happen at a specific image position or simultaneously in different image positions. For sake of simplicity, in the following we will not consider the contribution of the spatial accuracy, *a*.

The score for correct detections is defined as follows:

$$s_c(t,d) = \begin{cases} 0 & t \in \left]-\infty, B\right[, t \in \left]D, \infty\right[; \\ \dfrac{S(d)}{A-B}(t-B) & t \in [B, A]; \\ S(d) & t \in \left]A, 0\right[; \\ \dfrac{S(d)}{D}(D-t) & t \in [0, D]; \end{cases}$$

where *A*, *B* and *D* are constants and *S(d)* is a function with its maximum when the duration of the detected event is equal to the duration of the associated ground-truth event (*d=1*).

$$S(d) = \begin{cases} 50 * \left[2 - (1-d)^2\right] & d \in [0,2] \\ 50 & d \in \left]2, \infty\right[\end{cases}$$

The constant *A* represents the accepted anticipation of the detection of an event. The constant *B* represents the largest tolerated anticipation of the detection of an event. Finally, the constant *D* represents the largest tolerated delay for the detection of an event. The values of *A*, *B*, and *D* depend on the type of event. For instance, in case of a critical alarm event, *A = -1000 ms*, *B = -2000 ms*, *D = 1000 ms*, and s_p = *-5*, s_n = *-1000*.

In addition to s_r, the application-dependent evaluation takes into account the complexity of the solution. Complexity is here measured based on the number of parameters required by the algorithm, the spatial resolution and the temporal sampling step. When choosing between two algorithms with comparable s_r, the selected one is that using the smaller spatial resolution and larger temporal sampling step.

5. DISCUSSION

The comparison between different solutions can be achieved in a distributed or in a centralized fashion. In the former case, the same data set is distributed to the research community. The researchers are then expected to provide the results in the standardized format. The advantage of this approach is that software is not distributed. The disadvantage is that it is more difficult to guarantee a uniformity of protocol usage, unless a completely automated software solution exists. The second solution, the centralized comparison, requires that the researchers upload the executable of their algorithms to a central server. Then different algorithms are run in the same conditions. The advantage of a centralized solution is that it is possible to use datasets that otherwise cannot be distributed for privacy issues. However, researchers are not always happy to share their code.

Finally, performance characterization is a time consuming task. An important issue is therefore the automation of the testing process. In order to facilitate the testing, a clear evaluation protocol needs be defined. This protocol includes not only a set of benchmark criteria, as described in the previous sections, but also a standardized format to present the results using, for instance, XML.

6. CONCLUSIONS

We proposed a benchmarking protocol to evaluate and compare object tracking algorithms. The protocol aims at testing the algorithms per se as well as in the context of a specific application. In the specific application errors are weighted according to the particular task at hand. We provided the example of an evaluation protocol for an indoor surveillance scenario. We hope that this will serve as test-bed and allow for advancement of the area.

7. REFERENCES

[1] G. Pingali and J. Segen, “Performance evaluation of people tracking systems”, Proc. of 3rd IEEE Workshop on Applications of Computer Vision, pp. 33–38, December 1996.

[2] T. Schlogl, C. Beleznai, M. Winter, and H. Bischof. “Performance evaluation metrics for motion detection and tracking”, Proc. of IEEE Conf. on Pattern Recognition, v. 4, pp 519–522, August 2004.

[3] V.Y. Mariano, *et al*. “Performance evaluation of object detection algorithms”, Proc. of IEEE Conf. on Computer Vision and Pattern Rec., v.3, pp. 965–969, August 2002.

[4] C.J. Needham, R.D. Boyle, “Performance evaluation metrics and statistics for positional tracker evaluation”, Proc. of Int. Conf. on Computer Vision Systems, pp. 278–289, April 2003.

[5] J. Black, T.J. Ellis, and P. Rosin, “A novel method for video tracking performance evaluation” In Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, pp. 125–132, October 2003.

[6] C. Jaynes, S. Webb, R.M. Steele, and Q. Xiong. “An open development environment for evaluation of video surveillance systems”, IEEE Int. Workshop on Visual Surveillance and Performance Eval. of Tracking and Surveillance, June 2002.

[7] A. Cavallaro, O. Steiger, T. Ebrahimi, "Tracking video objects in cluttered background", IEEE Transactions on Circuits and Systems for Video Technology, April 2005.

[8] A. Cavallaro, E. Drelie, T. Ebrahimi, "Objective evaluation of segmentation quality using spatio-temporal context", Proc. of IEEE Int. Conf. on Image Processing, Rochester (NY, USA), September 2002.